EPORT DOCUMENTATION PAGE

2

1b RESTRICTIVE MARKINGS

3 DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release,
distribution unlimited

DECLASSIFICATION / DOWNGRADING SCHEDULE

PERFORMING ORGANIZATION REPORT NUMBER(S)
IIT-CE-R-90-25

5 MONITORING ORGANIZATION REPORT NUMBER(S)
AEOSR-TR- 91 0134

NAME OF PERFORMING ORGANIZATION
ssachusetts Institute of Tech.
pt. of Civil Engineering

6b OFFICE SYMBOL (If applicable)

7a NAME OF MONITORING ORGANIZATION
U.S. Air Force Office of Scientific Research

ADDRESS (City, State, and ZIP Code)
7 Massachusetts Avenue
ambridge, Massachusetts 02139

7b ADDRESS (City, State, and ZIP Code) Bld 410
Bolling Air Force Base
Washington D.C. 20332-6445

NAME OF FUNDING / SPONSORING ORGANIZATION
AFOSR and AFESC

8b OFFICE SYMBOL (If applicable)
N/A

9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
AFOSR-87-0260

ADDRESS (City, State, and ZIP Code) Bl 410
OSR - Bolling AFB, Washington D.C.20332-6448
ESC - Tyndall AFB, Florida 32404-6061

10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO
61102F	2302	C2	

TITLE (Include Security Classification)
ochastic and Centrifuge Modelling of Jointed Rock Vol. 3 Stochastic and Topological Fracture Geometry Model

PERSONAL AUTHOR(S)
Jun-Suk Lee; Daniele Veneziano, Herbert H. Einstein

a. TYPE OF REPORT
Final

13b. TIME COVERED
FROM 6/87 TO 5/90

14. DATE OF REPORT (Year, Month, Day)
1990, August, 31

15 PAGE COUNT
306

SUPPLEMENTARY NOTATION

COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Jointed (Fractured) Rock, Stochastic Modelling, Fracture Geometry

ABSTRACT (Continue on reverse if necessary and identify by block number)

A hierarchical model is developed to represent the geometry of fracture sets. The most important characteristics of this model are the possibility to model sequential fracture genesis as it usually occurs in nature, and to model clustering. This model uses the inhomogeneous or the doubly stochastic (Cox) point process model to represent the midpoints of the primary set. In a second step, the fracture trace lengths of the primary set are modelled. This is done using Maximum Likelihood Estimates (MLE) of the observed data. In

(Continued on next page.)

DISTRIBUTION / AVAILABILITY OF ABSTRACT
☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

NAME OF RESPONSIBLE INDIVIDUAL
JR S Wu

22b. TELEPHONE (Include Area Code)
202-767-6963

22c OFFICE SYMBOL
N/A

83 APR edition may be used until exhausted.
All other editions are obsolete.

Block 19 - continued

a third step, the secondary set is considered; in particular the independence/dependence of the two sets. The correlation of location with trace length is considered with the line-kernel function methods and the nearest neighbor fiber distance method.

The hierarchical model has been applied to two cases in which detailed fracture patterns have been observed, one with a single set and one with two sets. The validity of the model was checked by visual comparison between model prediction and mapped patterns and most importantly, by statistical tests such as the second-moment analysis and the Monte Carlo test. A satisfactory fit of the predicted pattern was obtained in both cases.

Finally, a topological application of the stochastic geometry model is developed. Using a fully persistent block model, the kinematic as well as the kinetic stability of a rock slope is investigated. Through sensitivity studies, the most significant parameters affecting slope stability are found.

It is important to note that the combined hierarchical and topological model can be applied to any problem involving stability or failure of fractured bodies or masses; thus not only slope stability but also tunnel stability can be considered. Although limited at present to fully persistent fractures, the model concept is also applicable to non-persistently fractured masses/bodies and can thus include concrete structures.

MIT CE R-90-25

STOCHASTIC AND CENTRIFUGE MODELLING OF JOINTED ROCK

Part III - Stochastic and Topological Fracture Geometry Model

Grant No. AFOSR-87-0260

Final Report 1990

Prepared by

Jun-Suk Lee
Herbert H. Einstein
Daniele Veneziano

Sponsored by

U.S. Air Force
Air Force Office of Scientific Research
Bolling Air Force Base
Air Force Engineering Services Center
Tyndall Air Force Base

September 1990

MIT

DEPARTMENT
OF
CIVIL
ENGINEERING

[illegible]
[illegible]
[illegible]

ACKNOWLEDGEMENTS

The research underlying this report was sponsored by the U.S. Air Force Office of Scientific Research and the U.S. Air Force Engineering and Services Center. The authors would like to acknowledge this support. In particular they would like to express their gratitude to Dr. S. Wu of AFOSR and Lt. S. Kuenen of AFESC for their encouragement and critical review of the work. This interaction went beyond the usual and was very beneficial to the project and to the authors. The authors would also like to thank Dr. P. Thompson formerly of AFESC for helping to initiate this research. Finally we would like to acknowledge the support by NSF through Supercomputer time at the J. v. Neumann Supercomputer Center.

Table of Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

Since rock masses are usually discontinua, one needs to consider fracture geometry when considering rock mass behavior. The MIT rock mechanics group has developed idealized stochastic models : Baecher, et al. (1977) assumed a disk shaped fracture in a space. The Baecher model can be constructed by generating a homogeneous Poisson point pattern for fracture centers and by adopting suitable fracture size distribution functions as well as fracture orientation distribution functions. Veneziano (1979), in his model, used a Poisson plane process in space followed by a Poisson line process in each plane. Using a persistence parameter, polygonal fractures can be constructed in space. Dershowitz (1985) modified the Veneziano model by considering fracture intersection behavior. With a Poisson plane process, he generates intersections of various fracture planes in space which in turn bound polygonal fractures. These simplified stochastic fracture models have been applied in a number of cases. They are limited, however, since they cannot represent many fracture patterns encountered in the field. Specifically :

- They do not account for spatial nonhomogeneities such as fracture clustering.
- The models are only loosely tied to the geologic genesis of the fractures. In particular, most models assume independence among fracture sets. From a geologic viewpoint, this assumption is often incorrect since the fracture sets are, in almost all cases, sequentially generated with inter-relations among themselves.
- Only in a few cases have the models been validated using actual fracture data.

In this research project, advanced stochastic geometry models are created which better represent reality.

Stochastic geometry can handle both the geometric and the stochastic nature of a structure (i.e., rock mass). In general, stochastic geometry can be considered in different ways :

1. Based on a geometry (shape), which can be a point or line process in a plane, or a three-dimensional Random Closed Sets (RACS) process.
2. Based on data sets for the different characteristics; for each characteristic, the data set can be univariate or multivariate processes.

Application of point processes can be found in the field of forestry. Trees in a forest can be represented as points in a region. From this simplified process, we can find the inter-relations among trees. Examples of line processes come from material science. A defect in the metal surface can be regarded as a line (or a fiber with finite length). In the field of microscopy, one often uses a RACS model where the closed sets consist of spheres or disks in a space. If the characteristics of the model are not uniform, one employs bivariate or multivariate point, fiber or RACS processes. For example, if two kinds of trees exist in a region, the bivariate point process becomes a plausible model.

Within the framework of rock fracture modeling, a point process can be used if the distribution of fracture traces on a rock surface is simple. Each trace is actually a segment (fiber) of finite length, but if the trace length, trace orientation and trace position are independently distributed, we can describe the segment as a point, usually the mid point. If the above mentioned trace characteristics are correlated with each other, we use the fiber process for modeling fractures in a plane. A RACS process seems to be best suited for the three dimensional situations we are actually interested in. However, these models rely on information which is not readily available such as size and shape of fractures. Eventually, RACS process models will have to be developed.

We, at this stage, want to concentrate on the point and fiber processes. Specifically, we will develop alternative models to the conventional homogeneous Poisson point and Poisson fiber processes. The main features of these models are the hierarchical description of fracture sets which allows one to reproduce the sequential genesis of fracture sets, and the consideration of dependencies among fractures of the same set or of different sets. The sequential generation and correlation of fracture sets correspond to what happens in nature. Equally important as the modelling principle is the availability of statistical procedures to estimate parameters and validate the model. These will provide the basis for future work on the RACS processes.

To apply the hierarchical fracture geometry model, a two dimensional rock slope will be considered. The fracture patterns simulated with the hierarchical model can be used in any rock mass problem such as flow through fractured rock masses, deformability of fractured rock masses and stability of fractured rock masses. The last of these problems is chosen here because it has great practical significance both regarding rock slope stability and stability of any fractured mass (rock mass around tunnel, concrete structures with cracks, etc). For such applications, existing models are not suitable. Consequently, new models need to be developed which, in our case, will consist of a combination of the hierarchical fracture model and a topological model.

In Chapter 2, we will discuss the basic characteristics of the point process and of some alternative models which may be used in rock fracture modeling. In particular, we will consider the Complete Spatial Randomness (CSR) as an indicator of the homogeneous, isotropic point process. In Chapter 3, the appropriate point process models which can be best fit to our mapped data are specified and developed. Additional mathematical expressions are introduced which make the unknown functions (in our case, the trace intensity function) easy to formulate. In Chapter 4, we will introduce the basic characteristics of the fiber process. Also traditional fiber process models will be reviewed and criticized. In Chapter 5, we will expand our point process model to a fiber process model. Topological application of the stochastic fracture geometry model is considered in Chapter 6. Finally, in Chapter 7, conclusions will be drawn as to advantages and limitations of the fracture geometry models.

Chapter 2

BASIC CHARACTERISTICS OF POINT PROCESSES

Point processes can be regarded as statistical models for point-like objects in a plane. Traditionally trees in a forestry or earthquake occurences in the space-time domain have been represented by point patterns. The reason for using a point processes is to analyze patterns of points and to suggest a plausible mechanism by which they might have been generated. In our case, the traces on a rock outcrop are considered as segments, and can be represented as points, usually mid points, in a plane. (This is a first approximation ; more complex processes will be introduced later). Fig. 2-1 is an example of a simple trace pattern and we will consider such trace data with point processes. Usually, one will assume a Poisson point process as a null hypothesis when generating center points of rock fractures in a plane or space.

In this Chapter, we, first of all, check the above mentioned null hypothesis using mapped data such as the fracture traces of Fig. 2-1. There are many tests which can be used to check randomness assumptions. Two relatively powerful tests are described here, and a brief history of various test methods is given in Appendix A. Next, the second moment properties of the point processes are considered since, in most cases, point patterns can be characterized by second moments. Finally, we will discuss various point process models, including the homogeneous Poisson point process model, which may be applied to rock fracture modelling.

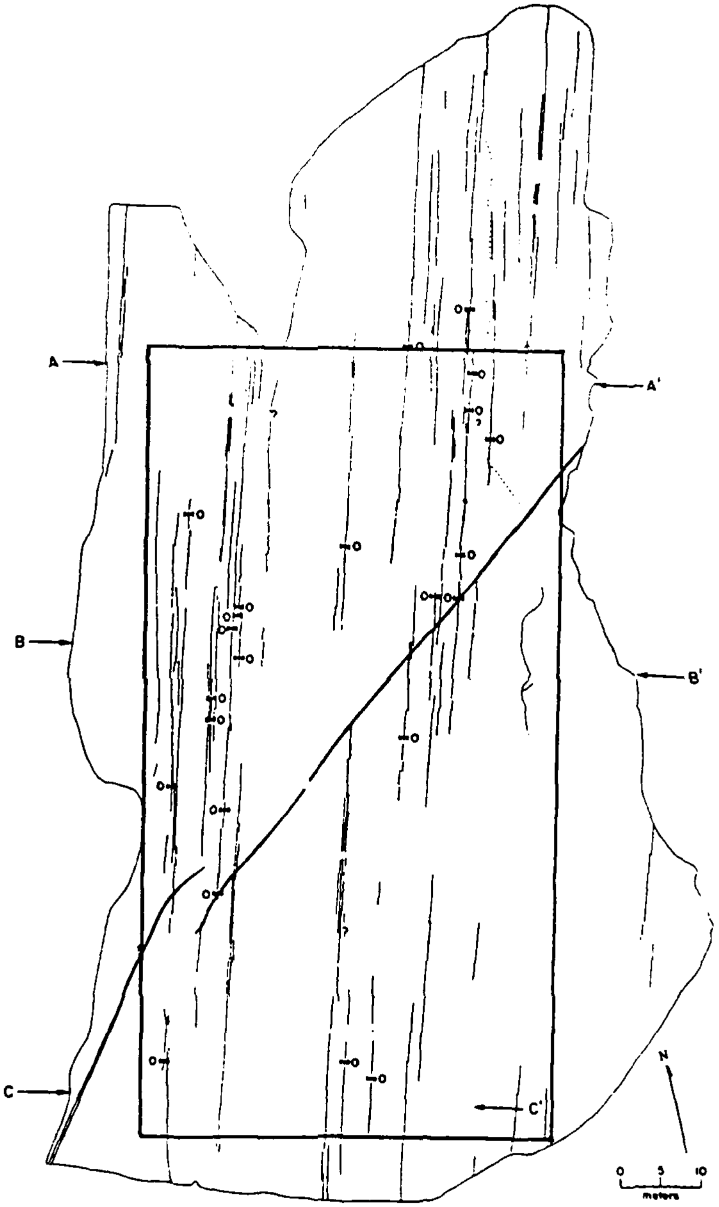


Figure 2-1: Map of Florence Lake Outcrop (After Segall & Pollard, 1983 ; Only Traces within Rectangular Region are Considered)

2.1 Test of Complete Spatial Randomness (CSR)

The simplest assumption of point pattern characteristics is no interaction between the points. This is called Complete Spatial Randomness (CSR) (Diggle, 1983). If the intensity of points changes from place to place, the pattern is not homogeneous and we call it inhomogeneous or heterogeneous. Directional patterns are called anisotropic.

The hypothesis of CSR for a point pattern asserts that,

1. The number of events in any planar region A with area $|A|$ follows a Poisson distribution with mean $\lambda|A|$, where λ is the intensity.
2. Given n events x_i in a region A, the x_i are an independent random sample from the uniform distribution on A.

There are two reasons for considering CSR.

1. CSR is a minimal prerequisite for randomness.
2. CSR operates as a dividing hypothesis between regular and aggregate patterns.

Therefore, if the given set of points does not follow CSR (because there is an attractive or inhibitive correlation among points), our aim will be to find non-Poisson models and to establish a method for testing the fit of any proposed model.

To test CSR of a given data set, we use two different methods. One is based on distance measures, the other uses a quadrat count. Only the distance measure method will be discussed here, specifically, inter-event distances and nearest neighbor distances. This will be done because, in some cases, the quadrat count method cannot provide much information, for example, on clustering or inhibitory behavior, and it is not easy to visualize. Furthermore, if the number of data points is small, the quadrat count method may not be useful. The specific distance measure methods described here are the inter-event distances and nearest neighbor distances.

2.1.1 Inter-event Distances

The theoretical distribution of the distance *t*, H(t), between two events independently and uniformly distributed in a region A depends on the size and shape of A, but can be expressed in closed form for the most common cases of a square or circular area A (Bartlett, 1964). Assuming that for the particular region A in question, H(t) is known, we can calculate the empirical distribution function (EDF) of inter-event distances. This function, $\hat{H}_1(t)$ say, represents the observed proportion of inter-event distances t_{ij} which are at most t. Thus,

$$\hat{H}_1(t) = \{\frac{1}{2}n(n-1)\}^{-1}\, no.(t_{ij} \leq t) \tag{2.1}$$

where *no.* means "the number of" and $\hat{H}_1(t)$ is the empirical distribution function. A plot of $\hat{H}_1(t)$ on the ordinate against H(t) on the abscissa should be roughly linear if the data are compatible with CSR. To assess the significance and the deviations from linearity, the conventional approach would be to find the sampling distribution of $\hat{H}_1(t)$ under CSR. Since this is complicated by the dependence between inter-event distances with a common end-point, one, therefore, proceeds as follows :

1. Calculate EDF $\hat{H}_i(t)$, i = 2,3,...,s, from each of s-1 independent simulations of n events independently and uniformly distributed on A
2. Define upper and lower simulation envelopes,

$$U(t) = \max\left(\hat{H}_i(t)\right),$$
$$L(t) = \min\left(\hat{H}_i(t)\right),\quad i = 2,3,\cdots,s. \tag{2.2}$$

   these simulation envelopes can also be plotted against H(t), and have the property that under CSR, and for each t,

$$P\left(\hat{H}_1(t) > U(t)\right) = P\left(\hat{H}_1(t) < L(t)\right) = s^{-1} \tag{2.3}$$

   since we assume that each simulation is distributed independently and uniformly on A. The simulation envelopes are intended to help in the interpretation of the plot of $\hat{H}_1(t)$ against H(t). That is, if the theoretical pattern lies between U(t) and L(t) throughout its range, it means acceptance of CSR.

3. If the region A is one for which the theoretical distribution function H(t) is unknown as in our case (rectangular region), a test can still be carried out if H(t) is replaced by

$$\overline{H}_i(t) = (s-1)^{-1} \sum_{j \neq i} \hat{H}_j(t) \tag{2.4}$$

Similarly, the graphical procedure then consists of plotting $\hat{H}_1(t)$, U(t) and L(t) against $\overline{H}_1(t)$ or, to make it easy to visualize, plotting $\hat{H}_1(t)$, U(t), L(t) and $\overline{H}_1(t)$ against distances. Note that because $\overline{H}_1(t)$ is a simulated value under the assumption of CSR and does not include the original data, it provides an unbiased estimate of H(t). Checking CSR with simulation envelopes can also be applied here, i.e., if $\overline{H}_1(t)$ lies between maximum (U(t)) and minimum (L(t)) envelopes throughout its range, it means acceptance of CSR.

Fig. 2-2 shows an inter-event distance test of our mapped pattern from Fig. 2-1, where the averaged value is given in Eqn (2.4), and for clarity, U(t) and L(t) are replaced by maximum and minimum value, respectively. From this figure, we do not accept the CSR assumption for the data of Fig. 2-1. This means, our mapped pattern is not random and there seems to be a certain interaction among mid points. However, with this test, it is not easy to see the degree of interaction. In other words, we see the clustered behavior of the traces from Fig. 2-1, but we cannot use this test to quantitatively describe the clustering behavior.

Also from a simple Monte Carlo test (Barnard, 1963), we can decide on the acceptance of CSR for given data. When we use the Monte-Carlo test, it provides a useful check on the applicability of the asymptotic theory. Let u_1 be the observed value of a statistic U and let u_i, $i = 2, \cdots, s$ be corresponding values generated by independent random sampling from the distribution of U under a simple hypothesis H. Let $u_{(j)}$ denote the j-th largest amongst u_i, $i = 1, \cdots, s$. Then, under H,

$$P\{u_1 = u_{(j)}\} = s^{-1}, \quad j = 1, \cdots, s \tag{2.5}$$

and rejection of H on the basis that u_1 ranks k-th largest or higher gives an exact, one-sided test of size $\frac{k}{s}$. For example, if u_1 is the highest among 20 sampling values, one rejects the hypothesis of the 5% significance level. This assumes that the values of the u_i are all

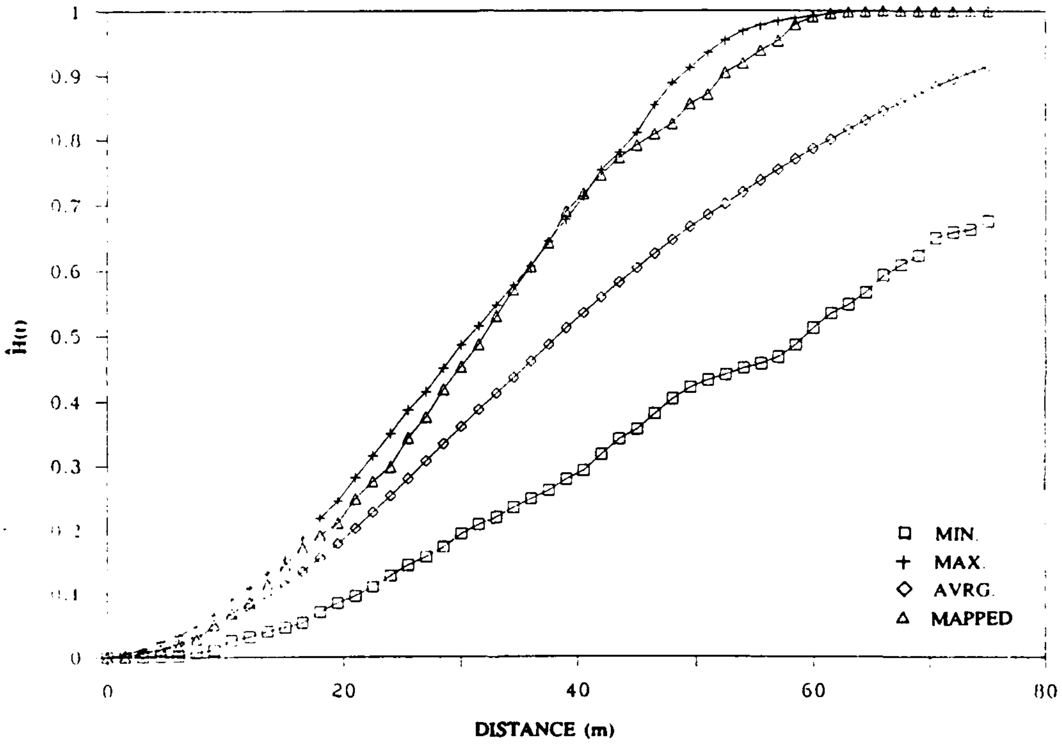


Figure 2-2: CSR test with Inter-event Distance

different, so that the ranking of u_1 is unambiguous. In our case, define u_i to be a measure of the discrepancy between $\hat{H}_i(t)$ and H(t) over the whole range of t,

$$u_i = \int \{ \hat{H}_i(t) - H(t) \}^2 dt \tag{2.6}$$

and proceed with the above mentioned Monte Carlo test based on the rank of u_1.

In our case, we tried 49 simulations (i.e., s = 50) and, from the one-sided test at the 2% significance level, we reject CSR for our mapped pattern.

2.1.2 Nearest Neighbor Distances

For n events in a region A, let y_i denote the distance from the i-th event to the nearest other event in A. The y_i are called nearest neighbor distances. We can calculate the EDF, $\hat{G}_1(y)$ say, of the nearest neighbor distances, i.e.,

$$\hat{G}_1 = n^{-1}\, no.(\, y_i \leq y\,) \tag{2.7}$$

In many practical situations, interactions between events exist, if at all, only on a small physical scale : for example, trees would be expected to compete for sunlight or nutrient within an area roughly confined to their crowns or root systems, respectively. In such a case, nearest neighbor distances provide an objective means of concentrating on small inter-event distances when a precise threshold distance cannot be specified in advance. The theoretical distribution of nearest neighbor distance Y under CSR depends on the number of events n and on A, and is not expressible in closed form because of complicated edge effects. An approximation which ignores these edge effects is obtained by noting that if $|A|$ denotes the area of A, then $\pi y^2 |A|^{-1}$ is the probability under CSR that an arbitrary event is within distance y of a specified event. Since the events are located independently, the approximate distribution function of Y is

$$G(y) = 1 - (\, 1 - \pi y^2 |A|^{-1}\,)^{n-1} \tag{2.8}$$

The EDF of nearest neighbor distances, $\hat{G}_1(y)$, can be compared with upper and lower simulation envelopes from simulated EDFs $\hat{G}_i(y)$: $i = 2, 3, \cdots, s$ in a similar way to the inter-event method. The approximate result, Eqn (2.8), can be used to suggest a suitable range of tabulation but, because it is approximaie, it is generally preferable to use the sample mean $\overline{G}_1(y)$ of simulated EDFs.

$$\overline{G}_i(y) = (s-1)^{-1} \sum_{j \neq i} \hat{G}_i(y) \tag{2.9}$$

Here again, $\overline{G}_1(y)$ involves only the simulation of CSR and it provides an unbiased estimate of G(t).

Now we can construct extreme envelopes like those in Fig. 2-3 in a similar way as we did in Fig. 2-2. We can plot $\hat{G}_1(t)$, U(t) (max.), L(t) (min.) and $\overline{G}_1(t)$ against distance t where U(t) and L(t) can be derived as follows,

$$\begin{aligned} &U(t) = \max\,(\hat{G}_i(t)), \\ &L(t) = \min\,(\hat{G}_i(t)), \quad i = 2,3,\cdots,s. \end{aligned} \tag{2.10}$$

Fig. 2-3 shows rejection of CSR. In addition, we clearly see the interaction of data points. Near the distance 1.0 and 3.0, we see the clustering phenomenon among traces, whereas near 2.5, there seems to be inhibitory interactions.

Again a Monte-Carlo test, which is analogous to Eqn (2.6), shows rejection of CSR at the 2% significance level. In this case, we replace the H function of Section 2.1.1. by G function shown above.

2.2 Second Moment Properties of Point Processes

To compare any candidate model with a set of data, we often use first and second moment properties of a point process. Here, E(x) is the expectation of a random variable x ; N(A) represents the number of events in a region A ; $|A|$ is the area of A ; dx is an infinitesimal region which contains the point x ; $b_x(t)$ represents the disc with center x and radius t ; R^2 is the entire plane where $R^2 = b(R) \times b(R) = \{ (x,y) : x, y \in R \}$ which is the co-ordinatization of the plane and generally R^d is referred to as the *d*-dimensional Euclidean space.

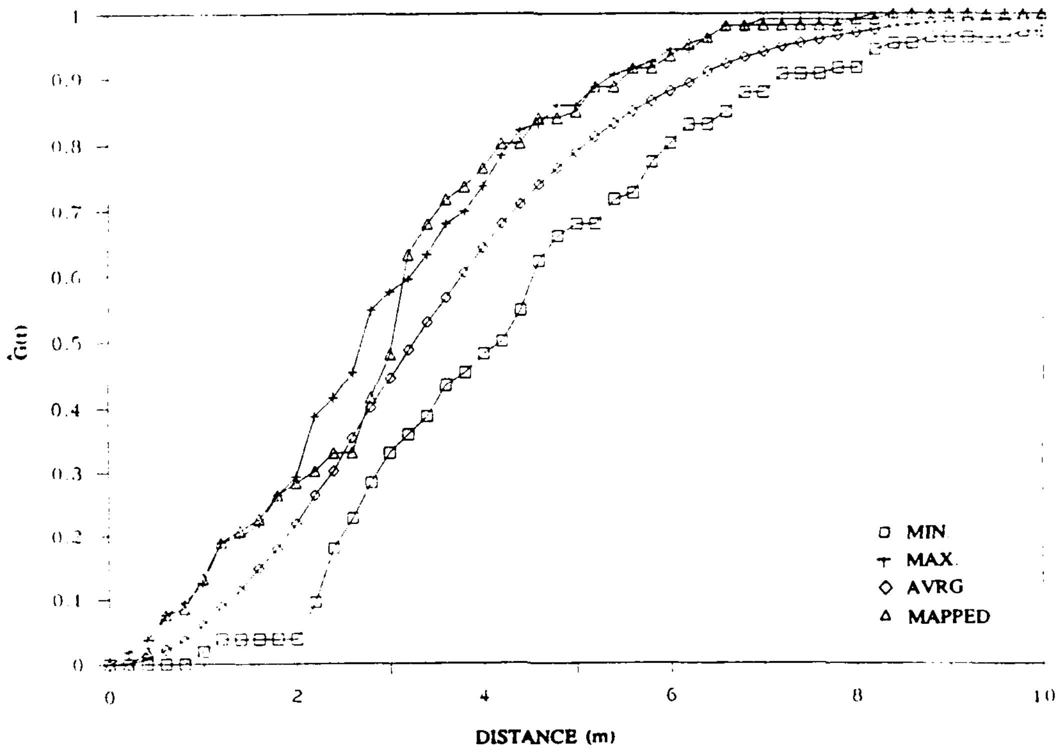


Figure 2-3: CSR Test with Nearest Neighbor Distance

First-order properties are described by an intensity function,

$$\lambda(\mathbf{x}) = \lim_{|d\mathbf{x}| \to 0} \left(\frac{E[N(d\mathbf{x})]}{|d\mathbf{x}|} \right) \tag{2.11}$$

For a stationary process (i.e., the process in any region A of the plane is invariant under arbitrary translation of A), $\lambda(\mathbf{x})$ assumes a constant value λ, the mean number of events per unit area. The second-order intensity function is similarly defined as

$$\lambda_2(\mathbf{x},\mathbf{y}) = \lim_{|d\mathbf{x}|,|d\mathbf{y}| \to 0} \left(\frac{E[N(d\mathbf{x})N(d\mathbf{y})]}{|d\mathbf{x}||d\mathbf{y}|} \right) \quad for\, x, y\ in\ \mathbf{R}^2 \tag{2.12}$$

For a stationary process, $\lambda_2(\mathbf{x},\mathbf{y}) = \lambda_2(\mathbf{x}-\mathbf{y})$; for a stationary, isotropic (i.e., invariant under rotation) process, $\lambda_2(\mathbf{x}-\mathbf{y})$ reduces further to $\lambda_2(t)$, where t is the distance between $\mathbf{x}$ and $\mathbf{y}$. In statistical mechanics, the scaled quantity $\lambda_2(t) / \lambda^2$ is referred to as the radial distribution function, although it is not a distribution function in the accepted statistical sense.

An alternative characterization of the second-order properties of a stationary, isotropic process is provided by the function K(t) (Ripley, 1977), which can be defined as

$$K(t) = \lambda^{-1} E[\, number\ of\ further\ events\ within\ distance\ \ t\ \ of\ \ an\ arbitrary\ event \,] \tag{2.13}$$

Thus, $\lambda^2 K(t)$ can be interpreted as the expected number of pairs of points less than distance t apart with the first point in a given set of unit area. In order to establish a link between K(t) and $\lambda_2(t)$, we shall assume that our process is orderly, by which we mean that multiple coincident events cannot occur. The expected number of further events within distance t of an arbitrary event is,

$$\begin{aligned} \lambda K(t) &= \int_0^{2\pi} \int_0^t \{ \lambda_2(x)/\lambda \}\, x\, dx\, d\theta \\ &= 2\pi\lambda^{-1} \int_0^t \lambda_2(x)\, x\, dx \end{aligned} \tag{2.14}$$

or

$$\lambda_2(t) = \lambda^2 (2\pi t)^{-1} K'(t) \tag{2.15}$$

where θ is the angle of the arc in a circle with center x and radius t. The covariance density γ is then

$$\gamma(t)=\lambda_2(t)-\lambda^2 \tag{2.16}$$

The above mentioned second moment characteristics play an important role when we consider the relative positions of two points of the process and quantify the apparent deviation from Poisson randomness to clustering or inhibition between points. For this, we, again, test CSR of our pattern with the second moment properties. If the mapped pattern follows the homogeneous Poisson point process, and if we use Monte-Carlo simulations, edge-corrected second moments of given data sets would exist between extreme envelopes. The extreme envelopes can be plotted by adopting a maximum and minimum $\hat{K}(t)$ value of the null hypothesis, and edge-corrections can be performed either by graphical concepts (Ripley, 1977 or Diggle, 1983) or by toroidal shifts (Lotwick & Silverman, 1982). The idea of the edge-correction can be understood, if we think of small areas near the boundaries, i.e., there may exist some additional points which do not lie inside the region, but the influence that these points have needs to be considered. We will discuss edge-correction methods in Appendix B.

Fig. 2-4 shows a second moment analysis of data given in Fig. 2-1. It is constructed by assuming that the data follow a homogeneous Poisson point process ; the simulation envelopes are the realizations of the homogeneous Poisson point pattern. From the figure, we see the discrepancies of the second moments between the simulated and the mapped values. As a result, we conclude that the mapped pattern does not follow the homogeneous Poisson point pattern, i.e. CSR. This was confirmed previously when we tested CSR with the inter-event and the nearest neighbor distance method. Also the Monte-Carlo test shows rejection of the homogeneous Poisson point process.

In some cases, we need to include the relative positions of three and more points ; Hanisch (1983) considers this case in detail. He shows that the third moment measure is determined by a function $K(t_1,t_2,\theta)$ of three variables, where t_1, t_2 are distances of two points from the origin and θ is the angle between these two points. Meanwhile, Baddeley &

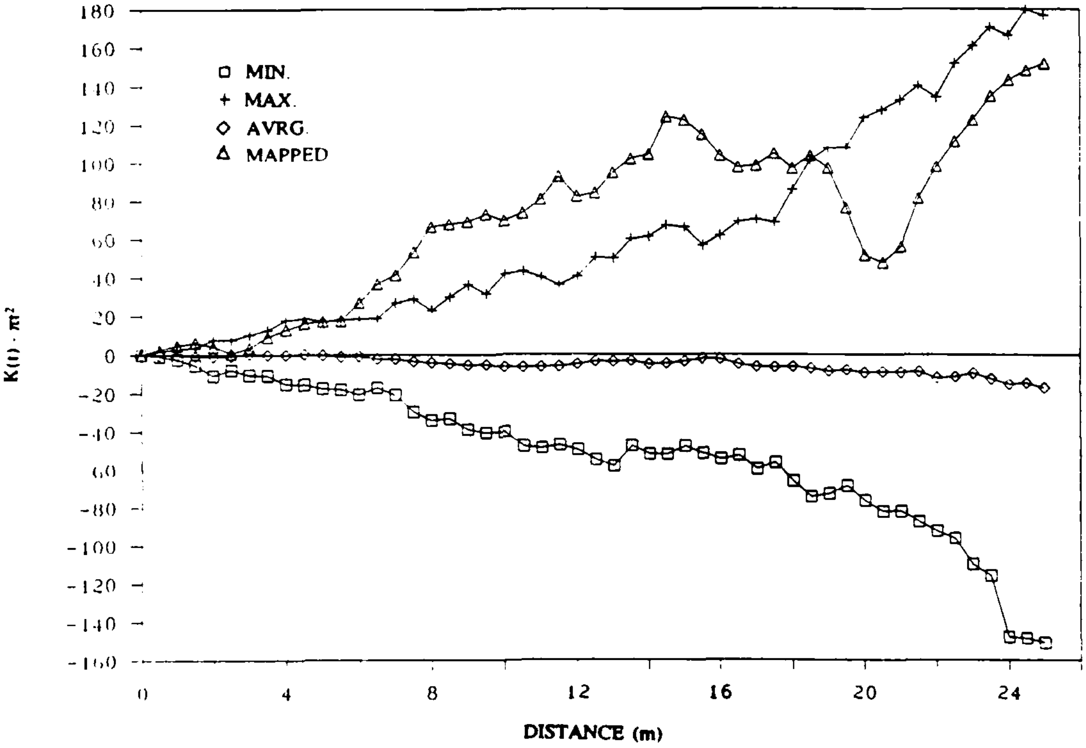


Figure 2-4: Second-order Moment Analysis of Mapped Data with Homogeneous Poisson Point Process Assumption

Silverman (1984) showed that investigation of the second moment properties alone cannot, sometimes, detect differences between point patterns with identical K functions.

2.3 Point Process Models

In this section, we discuss various models which can be applied to rock fracture modeling. We already mentioned the application of the homogeneous Poisson point process and saw that it is often inadequate to model rock fracture patterns. In order to provide a systematic treatment, the discussion below will nevertheless start with the definition of the homogeneous Poisson point process.

2.3.1 Homogeneous Poisson Point Process

This process represents the simplest possible stochastic mechanism for the generation of a spatial point process. It corresponds exactly to the definition of CSR given in Section 2.1.

1. For a given intensity $\lambda > 0$ and any finite planar region A, N(A) has a Poisson distribution with mean $\lambda|A|$.
2. Given N(A) = n, the n events in A form an independent random sample from the uniform distribution on A.
3. For any two different regions A and B, N(A) and N(B) are independent.

so from Eqn (2.12),

$$\lambda_2(t)=\lambda^2, \quad t>0 \tag{2.17}$$

and from Eqn (2.14) and Eqn (2.17),

$$K(t)=\pi t^2, \quad t>0 \tag{2.18}$$

This result is quite clear when we refer to the definition of the second moment function (see, Eqn (2.13)) since, in the homogeneous case, the expected number of further events within distance t become $\lambda\pi t^2$, thus leading to Eqn (2.18). Fig. 2-5 shows an example of the homogeneous Poisson point pattern.

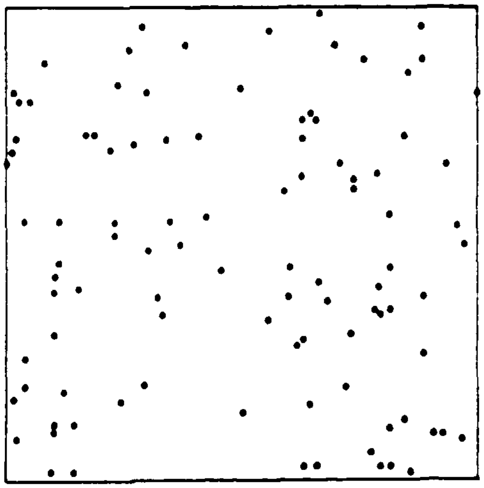


Figure 2-5: An Example of the Homogeneous Poisson Point Pattern

2.3.2 Inhomogeneous Poisson Point Process

A class of non-stationary point processes is obtained if the constant intensity λ of the Poisson point process is replaced by a variable intensity function $\lambda(x)$, where,

1. N(A) has a Poisson distribution with mean $\int_A \lambda(x)dx$.
2. Given N(A) = n, the n events in A form an independent random sample from the distribution on A with probability density function proportional to $\lambda(x)$.

However, difficulties remain when the intensity function, $\lambda(x)$, is hard to obtain. Fig 2-6 shows a realization of the inhomogeneous Poisson point pattern where $\lambda(x_1,x_2) = \exp(-2x_1-x_2)$.

2.3.3 Poisson Cluster Process

The Poisson cluster process (Neyman & Scott, 1958) is often called parent-daughter model. This kind of process provides a satisfactory basis for the modeling of aggregated planar point patterns.

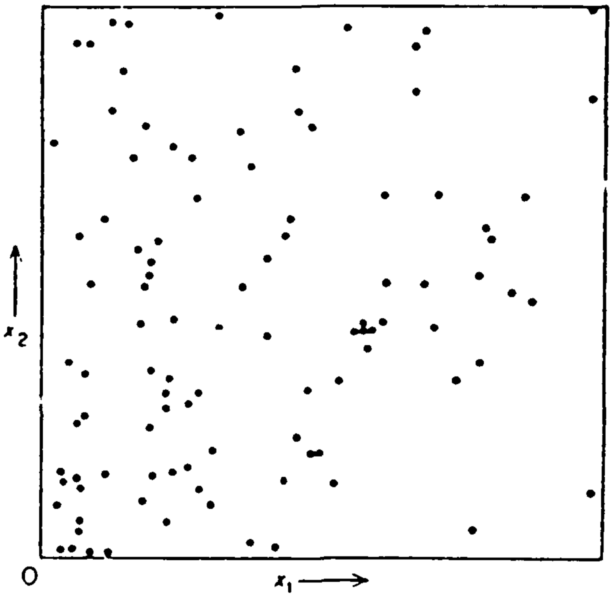


Figure 2-6: An Example of the Inhomogeneous Poisson Point Pattern

1. Parent events form a Poisson process with intensity ρ.
2. Each parent produces a random number s of offspring, realized independently and identically for each parent according to a probability distribution { P_s, s = 0,1,... }.
3. The positions of the offspring relative to their parents are independently and identically distributed according to a probability density function (PDF) *h*(.).

Poisson cluster processes are stationary, with intensity λ = ρE[s]. They are isotropic if *h*(.) is a radially symmetric function such as a bivariate normal distribution function. To express the second-order properties, it is convenient to define

$$h_2(z)=\int h(x)h(x-z)dx \qquad (2.19)$$

the PDF of the vector difference between two offsprings from the same parent. The expected number of ordered pairs of such offspring is $E[S(S-1)]$, so that

$$\lambda_2(x-y)=\lambda^2+\rho E[S(S-1)]\,h_2(x-y) \qquad (2.20)$$

in which the first term arises from a consideration of two offsprings at x and y from different parents, and the second from two offsprings of the same parent. Fig. 2-7 illustrates an example of a Poisson cluster process with 25 parents and four offsprings per parent.

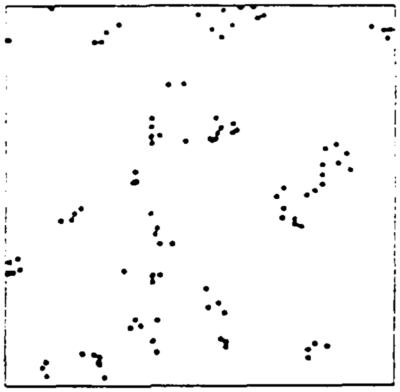


Figure 2-7: An Example of a Poisson Cluster Process

2.3.4 Cox Process (Doubly Stochastic Process)

The Cox process may be appropriate if the observed pattern reflects underlying environmental variation. The source of the environmental heterogeneity might itself be stochastic in nature. This suggests investigation of a class of doubly stochastic processes formed as inhomogeneous Poisson processes with stochastic intensity function. Thus,

1. $\{\Lambda(x); x \in R^2\}$ is a non-negative-valued stochastic process.
2. Conditional on $\{\Lambda(x)=\lambda(x); x \in R^2\}$, the events form an inhomogeneous Poisson process with intensity function $\lambda(x)$.

The difference between the Cox process and the inhomogeneous Poisson point process is that the intensity function of the Cox process is a stochastically varying function, whereas the intensity function of the inhomogeneous Poisson point process is deterministic.

First- and second-order properties are obtained from those of the inhomogeneous Poisson process by taking expectations with respect to $\{ \Lambda(x) \}$. Thus, in the stationary case, the intensity is,

$$\lambda = E[\Lambda(x)] \tag{2.21}$$

Also, the conditional intensity of a pair of events at x and y given { $\Lambda(X)$ } is $\Lambda(x)\Lambda(y)$, so that,

$$\lambda_2(x,y) = E[\Lambda(x)\Lambda(y)] \tag{2.22}$$

In the stationary, isotropic case, this can be written as

$$\lambda_2(t) = \lambda^2 + \gamma(t) \tag{2.23}$$

where,

$$\gamma(t) = Cov\{\Lambda(x), \Lambda(y)\} \tag{2.24}$$

and t is the distance between x and y. Note that the covariance function $\gamma(t)$ of the intensity process is also the covariance density of the point process. From a statistical point of view, the distinction between clustering and heterogeneity can only be made if additional information is available and Diggle (1977) suggested a two-phase test for distinction. Fig. 2-8 and Fig. 2-9 show a realization of the Cox process and corresponding covariance function, respectively (see Matern, 1971).

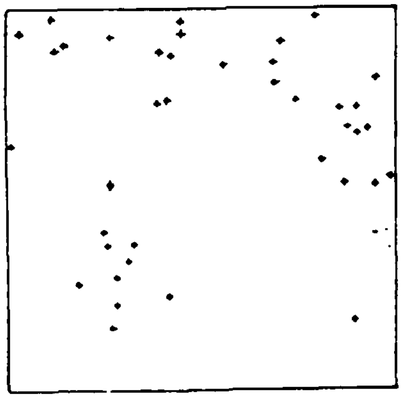


Figure 2-8: An Example of the Cox Process

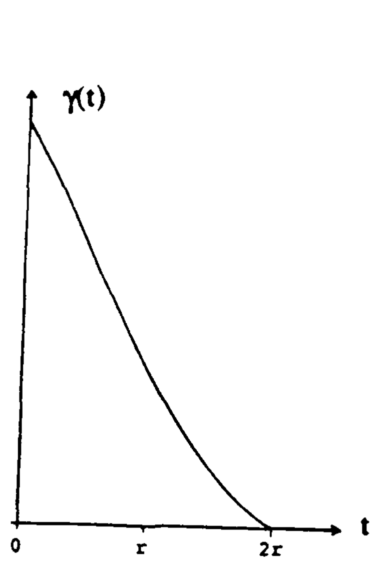


Figure 2-9: Covariance Density of the Process Used in Fig. 2-8

2.3.5 Simple Inhibition Process [Hard-core Process]

The alternatives to the Poisson processes described in Section 2.3.3. and 2.3.4. share a tendency to produce aggregated patterns. Regular patterns arise most naturally by the imposition of a minimum permissible distance, δ, between any two events. For example, as Matern (1986) described, assuming a random point representing a cell, specifically the core of the cell, then each point must be surrounded by a certain region where no other point can exist. Processes of this sort, which incorporate no further departure from complete spatial randomness, will be called simple inhibition (or repulsion) processes (or hard-core processes). The packing intensity of a simple inhibition process is defined as :

$$\tau=\lambda\pi\delta^2/4 \quad (2.25)$$

where λ is the intensity. Thus, τ is the proportion of the plane covered by non-overlapping discs of diameter δ, or the expected proportion of coverage for a finite region A.

Matern (1986) described two types of simple inhibition processes,

1. A Poisson process ρ is thinned by the deletion of all pairs of events a distance less than δ apart. That is, if two events have a mutual distance less than δ, both of them are deleted. The probability that an arbitrary event survives is therefore exp($-\pi\rho\delta^2$), and the intensity of the simple inhibition process is,

$$\lambda=\rho\, exp(-\pi\rho\delta^2) \tag{2.26}$$

2. In case of a dynamic scheme, the events of a Poisson process are marked with times of birth and an event is removed if it lies within a distance δ of an older event. An expression analogous to Eqn (2.26) can be obtained, but only by ignoring any consideration of whether or not the older event in question has itself previously been removed.

Recognition of this last aspect leads to a simple sequential inhibition process, defined on any finite region A as follows. Consider a sequence of n events x_i in A and let d(x,y) denote the distance between two points x and y. Then,

1. x_i is uniformly distributed in A.
2. Given $\{ x_i = x_j, j=1, \cdots, i-1 \}$, x_j is uniformly distributed on the intersection of A with $\{ y ; d(y,x_j) \geq \delta, j=1, \cdots, i-1 \}$.

Simple sequential inhibition is parameterized most naturally by its packing intensity, $\tau=n\pi\delta^2/4A$, where n is the number of events. Note that if too high a value of τ is prescribed, the sequential procedure may terminate prematurely. The maximum attainable packing intensity is a random variable whose distributional properties appear to be largely intractable. Stoyan & Stoyan (1985) modified Matern's two models by considering random hard-core distances. Fig 2-10 shows an example of the simple inhibition process.

2.3.6 Thinned Process

A thinned point process is defined by a primary point process $\{ N_o(dx) \}$ and a thinning field $\{Z(x)\}$, which is a stochastic process with realized values $0 \leq Z(x)) \leq 1$ for all x. Given realization of $\{ N_o(dx) \}$ and $\{ Z(x) \}$, the events x_i of $\{ N_o(dx) \}$ are retained, independently, with probabilities $z(x_i)$, and the corresponding realization of the thinned point process $\{ N(dx) \}$ consists of the retained events of $\{ N_o(dx) \}$.

The second-order properties of $\{ N(dx) \}$ are obtainable from those of $\{ N_o(dx) \}$ and $\{ Z(x) \}$. In particular, in the stationary, isotropic case, let p and $\gamma(t)$ denote the mean and covariance function of $\{ Z(x) \}$. Then the second-order intensity function of $\{N(dx)\}$ is

$$\lambda_2(t)=\lambda_{02}(t)\{ \gamma(t)+\rho^2 \} \tag{2.27}$$

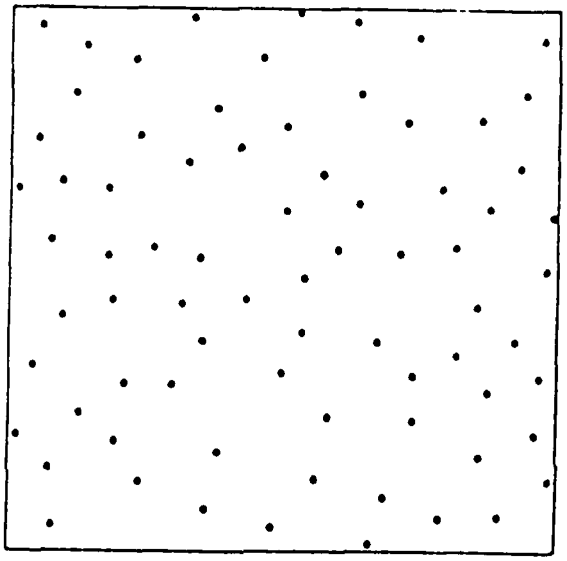


Figure 2-10: An Example of the Simple Inhibition Process

where $\lambda_{02}(t)$ is the corresponding second-order intensity function of { $N_o(dx)$ }. The K-function of { N(dx) } and { $N_o(dx)$ } are therefore related by

$$K(t)=K_o(t)+p^{-2}\int_0^t \gamma(u)K'_o(u)\,du \qquad (2.28)$$

a pattern of the thinned process is shown in Fig. 2-11.

Stoyan (1979) showed the relationship between the thinned process and the Cox process and derived radial distribution functions for various cases.

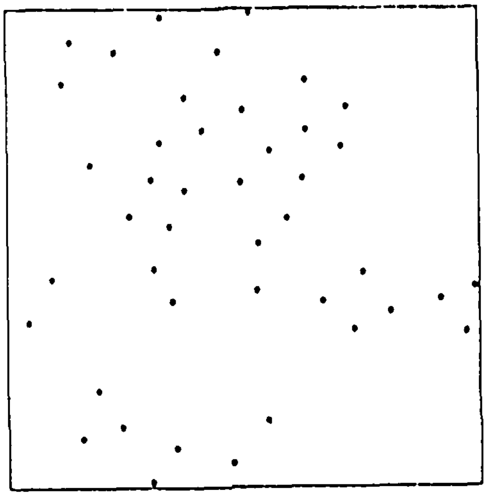


Figure 2-11: An Example of the Thinned Process

2.3.7 Soft-core Process

If there is an interaction between the points, then, it may not be "hard" or abruptly ending, but "soft" or continuously decreasing. In contrast to the hard-core process (i.e., inhibition process), the interaction of points is such that the inhibitory forces increase continuously with decreasing inter-point distance (Stoyan, 1987). Let a stationary Poisson process of intensity λ_h be given. Its points get independent marks which are distributed according to the continuous distribution function F on [0,∞). Then the initial point process is thinned as follows. A point x with mark r(x) survives if and only if there is no other point y with mark r(y) such that ;

1. x is in the circle centered at y with radius r(y)
2. r(y)≥ r(x).

If F is discontinuous, then the points get an additional independent mark u uniformly distributed on [0, 1]. If r(x) = r(y), then condition 2. alone is replaced by u(x) ≥ u(y). Of course, the thinned point process is stationary and isotropic. Its intensity λ is given by

$$\lambda=\lambda_b p_r \tag{2.29}$$

where p_r is the probability that a point of the basic Poisson process is retained. It satisfies

$$p_r=\int_0^1 exp(-\lambda_b V((s))ds \tag{2.30}$$

Here V(s) is the volume of the three-dimensional set M(0,s), where for x = (x_1, x_2)

$$M(x,s)=\Big(\,(\xi,\eta,\zeta);\sqrt{(\xi-x_1)^2+(\eta-x_2)^2}\ \le\ F^{-1}(\zeta),\\ s\le\zeta\le 1\Big) \tag{2.31}$$

F^{-1} denotes the inverse of F. The product-density $\rho(r) = \lambda^2 g(r)$ is given by

$$\rho(r)=\lambda_b^2 k(r) \tag{2.32}$$

where $k(r)$ is the probability that both members of a point-pair of distance r of the basic Poisson Process are retained. It satisfies

$$k(r)=2\int_0^1\int_{T(s)} exp(-\lambda_b V(r,s,t))dtds \tag{2.33}$$

Here V(r,s,t) is the volume of M(0,s) ∪ M(r,t), where **r** = (r_1, r_2) is a point of $\mathbf{R}^2$ with distance r to origin 0. T(s) is the set of all real numbers t in [s,1] with (r_1, r_2, t) ∉ M(0,s).

Fig. 2-12 is a pattern of beadlet anemones on a rock in which the positions and sizes of the anemones are indicated. Stoyan (1987) used a soft-core process when he analyzed this pattern.

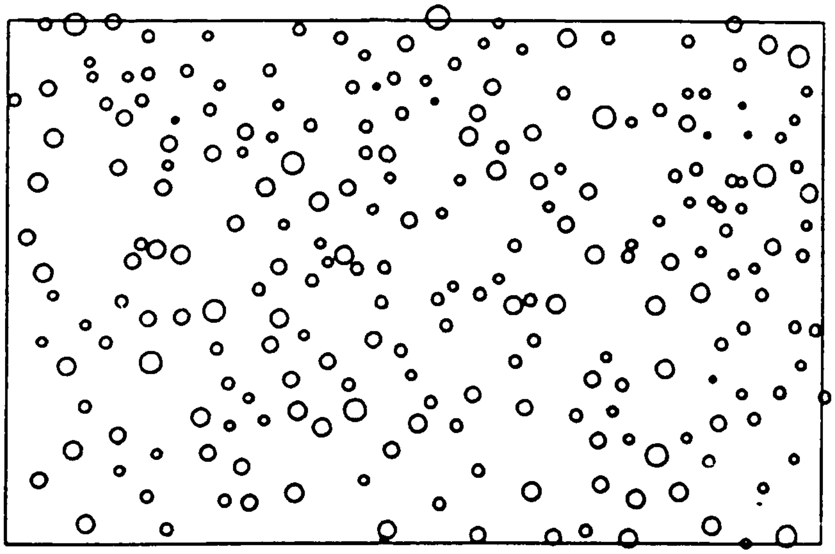


Figure 2-12: An Example of the Soft-core Process, After Upton & Fingleton (1985)

2.3.8 Discussion

We showed that our mapped pattern (Fig. 2-1) does not follow the CSR assumption using both the inter-event distance test and the nearest neighbor distance test. We checked it with the second moment measure and it also shows rejection of a CSR. To find a proper model, we discussed various possible point process models, including a homogeneous Poisson point process model, which can be applied to rock fracture modelling.

If we think of the large scale, the rock fracture model based on point processes can be perceived as stationary (Ripley, 1988). In such a case, CSR would probably satisfied. Ho'vever, if we confine our interests to a rather small portion of the rock surface, the pattern is probably nonstaticnary. It is in situations like this where the one of the nonhomogeneous models, which were discussed above, will have to be used. We will now examine if some of these models can represent patterns such as traces in Fig. 2-1.

Chapter 3

APPLICATION OF POINT PROCESSES TO ROCK FRACTURE MODELING

3.1 Introduction

Many fracture modeling studies choose a homogeneous Poisson point process as a basic assumption when generating mid-point patterns. In Chapter 2, we tested the randomness of the mapped pattern (Fig. 2-1) with three methods and concluded that it did not follow the homogeneous Poisson point process. Hence, we need to find an alternative model which has non-random properties. In our case, the inhomogeneous Poisson point processes and doubly stochastic point (Cox) processes seem to be suitable alternatives even though no clear rule exists as to which of these alternatives should be chosen.

In the following, we will consider the above mentioned two alternative models and discuss fitting the proposed model to the mapped data.

3.2 Inhomogeneous Poisson Point Model

As discussed in detail in Section 2.3.2, when using the inhomogeneous Poisson process, we should know an intensity function $\lambda(x,y)$ in advance. If we have an intensity function $\lambda(x,y)$, we can generate a point pattern with an Acceptance / Rejection (A/R) scheme as proposed by Lewis & Schedler (1979). This A/R scheme consists of simulating a Poisson process on a region A with intensity λ_o equal to the maximum value of $\lambda(x)$ within A and retaining an event at x with probability $\lambda(x)/\lambda_o$. The A/R scheme plays an important role when we generate a point pattern, especially a non-homogeneous pattern.

Fig. 3-1 is a mid point representation of our trace map of Fig. 2-1. Considering such a pattern, it is hard to find the intensity function λ(x,y). With regression, we can only find the y-directional intensity function like $\lambda(y) = \exp(0.36y - 0.028y^2 - 3.25)$ since significant clustering as well as inhibitory behavior exists simultaneously in the x-direction. To derive an x-directional intensity function, we use a kernel function *f*. Usually *f* is a symmetric probability density function, for example the normal density function, with the conditions,

$$\begin{aligned} &f(x) = f(-x) \\ &\int_{-\infty}^{\infty} f(x)\,dx = 1 \\ &\int_{-\infty}^{\infty} f(x)x^2\,dx = 1 \end{aligned} \qquad (3.1)$$

The kernel estimator with kernel function *f* is defined by

$$\hat{\lambda}(\mathrm{x}) = \frac{1}{nh}\sum_{i=1}^{n} f\left(\frac{\mathrm{x}-X_i}{h}\right) \qquad (3.2)$$

where *h* is the window width and often called the smoothing parameter or bandwidth.

The intuitive properties are as follows. Just as the moving average (see Silverman (1986) for details) can be considered as the height of the histogram (i.e., frequency) centered at the observation, the kernel estimator is a sum of bumps, where the window width *h* determines their width. This property of the kernel function, therefore, makes an intensity variation easy to derive. Fig. 3-2 shows a schematic representation of the kernel estimates. Provided the kernel *f* is everywhere non-negative and satisfies the condition, Eqn (3.1), it will follow at once from the definition that $\hat{\lambda}$ will itself be a probability density. Furthermore, $\hat{\lambda}$ will inherit all the continuity and differentiability properties of the kernel *f*, so that if *f* is the normal density function, then $\hat{\lambda}$ will be a smooth curve having derivatives of all orders. Particularly, one of the edge-corrected kernel estimators of λ(r) with efficient kernel function f_δ is (see Fiksel, 1988)

$$\hat{\lambda}_\delta(r) = \sum_{x}\sum_{y \neq x} \frac{f_\delta(|x-y|-r)}{2\pi|x-y|a(W_x \cap W_y)} \qquad (3.3)$$

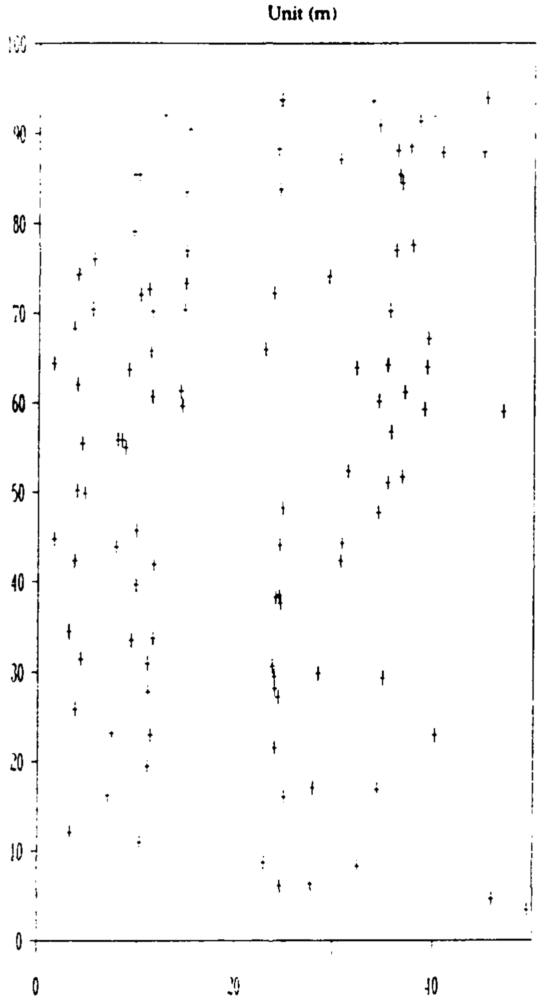


Figure 3-1: Mid Point Representation of Traces of Florence Lake Outcrop

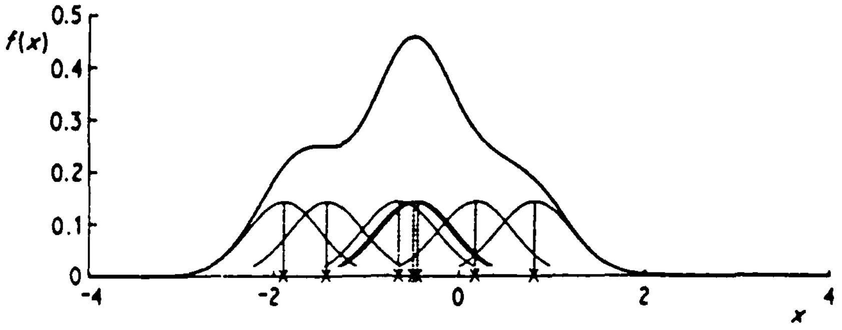


Figure 3-2: An Example of Schematic Kernel Estimate with Window Width 0.4

where, the summation goes over all points x and y of the pattern in the window of observation W. | | is the Euclidean distance ; f_δ is a kernel function ; W_x is a set of all points z of the plane having the form z = w + x, w ∈ W ; a = area ; δ is a band-width parameter ; r is a distance between two points and

$$f_\delta(r)=\frac{3}{4\sqrt{5}\,\delta}\left(1-\frac{r^2}{5\delta^2}\right), \quad -\delta\sqrt{5}\le r\le \delta\sqrt{5}$$

$$0 \qquad \textit{otherwise} \qquad (3.4)$$

The above kernel function is a general form of the Epanechnikov kernel (Epanechnikov, 1969). Though efficient, it is not easy to implement. However, a kernel estimator using normal distribution function provides a good estimate of the x-directional intensity function λ(x). Our edge-corrected kernel function is shown in Fig. 3-3.

We now have the intensity function both in the x- and y- direction. Hence, we can

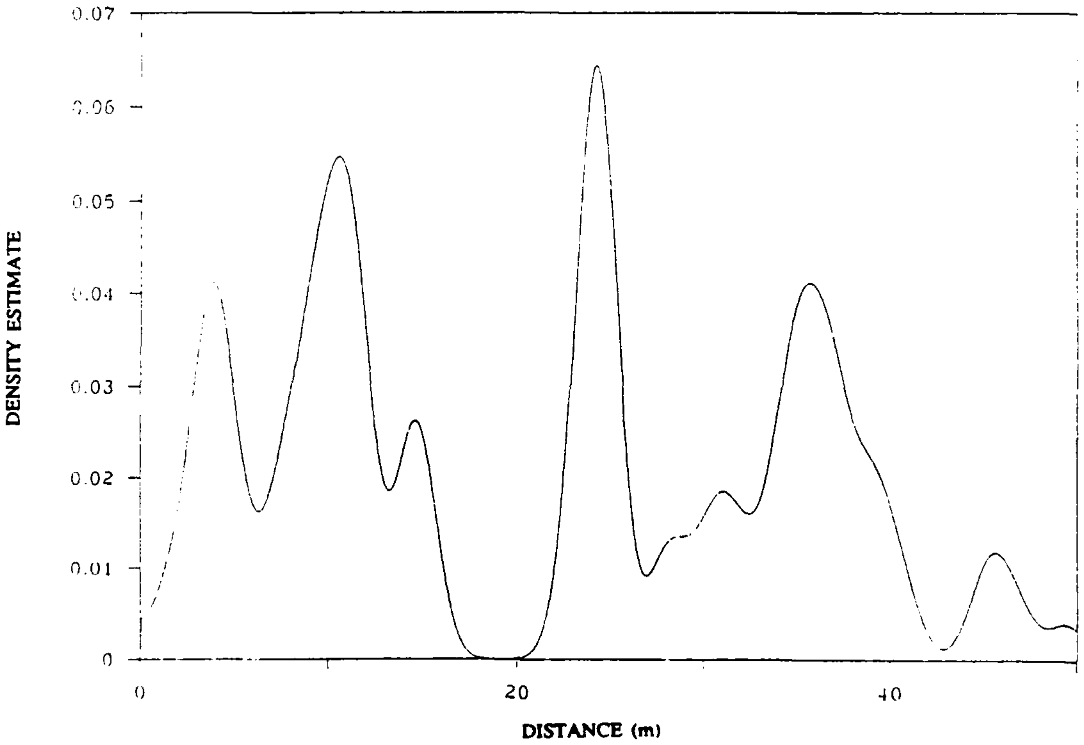


Figure 3-3: Kernel Estimator using Normal Distribution Function with Standard Deviation = 1.0

simulate the inhomogeneous Poisson point pattern with the aforementioned A/R scheme. Fig. 3-4 is one of the simulations of the inhomogeneous Poisson point process assuming a normal distribution as a kernel function. Comparing this simulated inhomogeneous Poisson point pattern with our mapped pattern (Fig. 3-1), we see some similarity. To check the inhomogeneous Poisson point process with our mapped data, we use the inter-event distance, nearest neighbor distance and second moment procedure (see Fig. 3-5 to Fig. 3-7). From these figures, we see good correspondence of our mapped pattern with the inhomogeneous Poisson point process. Again Monte-Carlo tests for each the above three procedures show satisfactory results, also.

3.3 Marked (or Weighted) Point Process

Before proceeding to the doubly stochastic point process, we introduce the marked (or weighted) point process. Fig. 3-8 illustrates an example of a marked point process where a point can be further characterized either by an ×-mark or by an o-mark. Since a point process itself does not give us any information on trace lengths or orientations, a marked point process can be used when considering additional characteristics of fractured rock if these are not too complicated. These characteristics can be included if we give marks or weights to each point assuming that the mark represents a trace length or orientation.

Considering mark-correlation functions and the cross-mark-correlation functions, which are similar to the second-moment properties, we can obtain information on marks such as trace length correlations and orientation correlations. Assume that there are two marks for each point such as the l-mark for a trace length and the m-mark for an orientation. Now calculate the mean of the product of the l-mark of the point in one of the infinitesimal areas and of the m-mark of that in the other one. This mean, denoted by $E_{lm}(r)$, is written as,

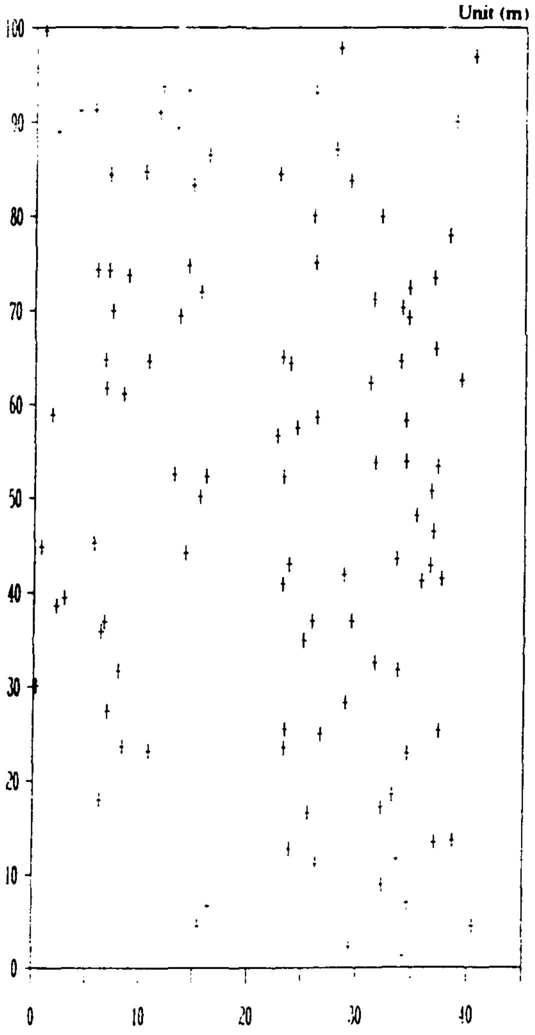


Figure 3-4: Inhomogeneous Poisson Point Pattern

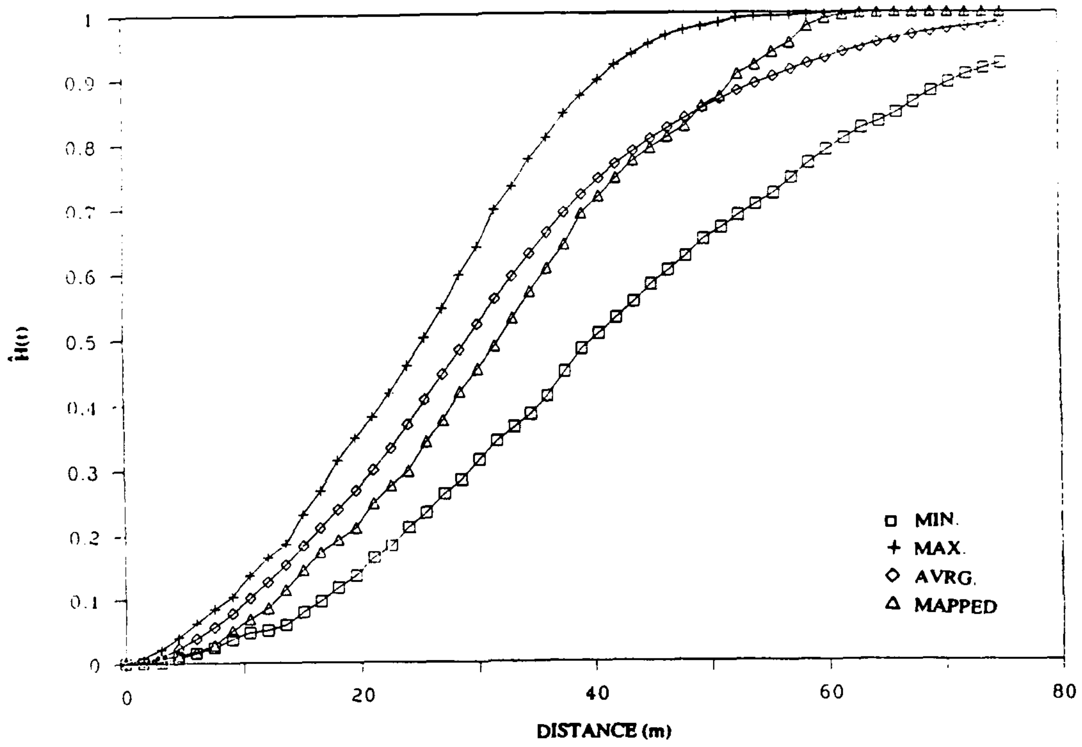


Figure 3-5: Inter-event Analysis of Inhomogeneous Poisson Point Process

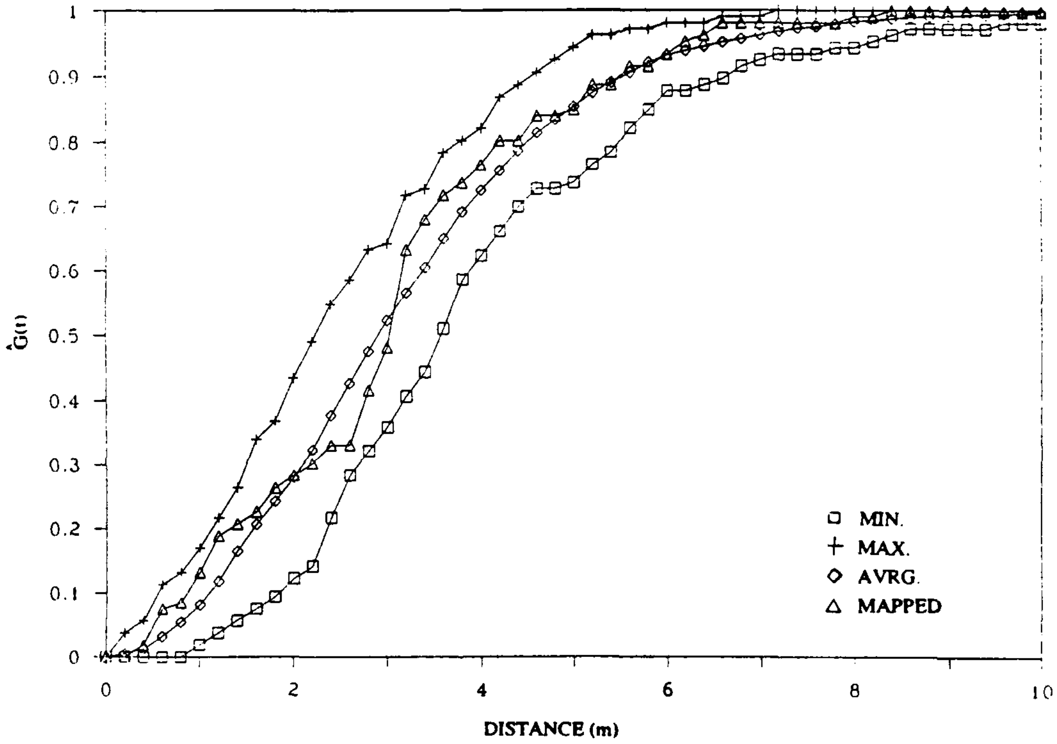


Figure 3-6: Nearest Neighbor Distance Analysis of Inhomogeneous Poisson Point Process

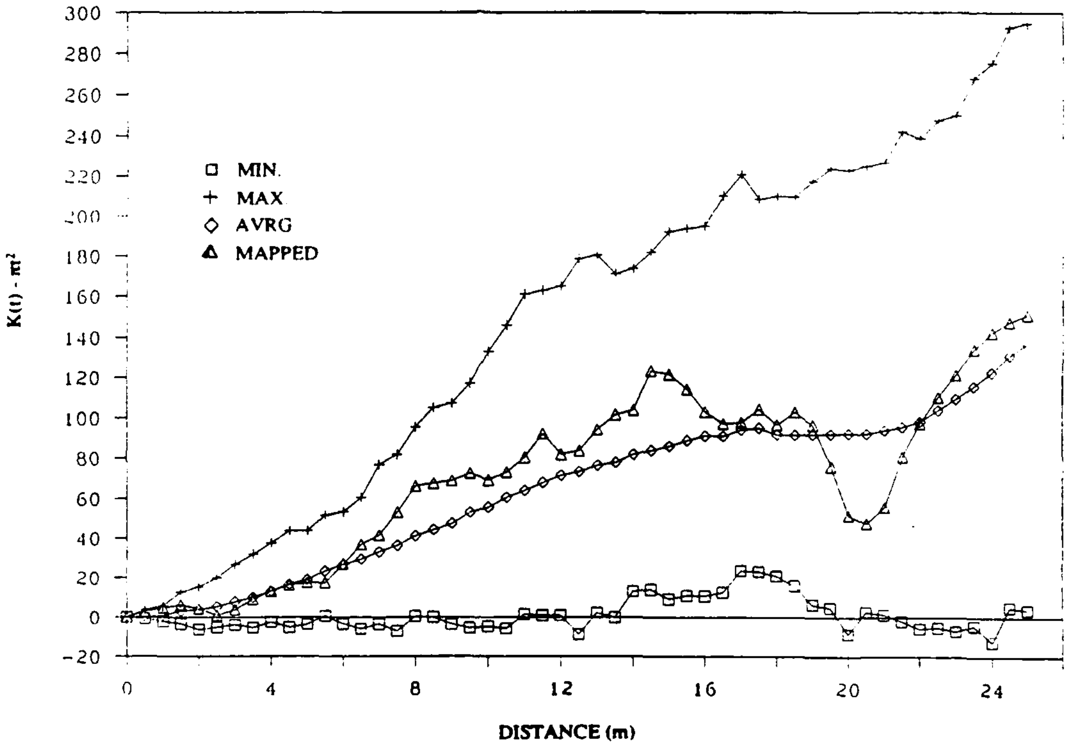


Figure 3-7: Second-moment Analysis of Inhomogeneous Poisson Point Process

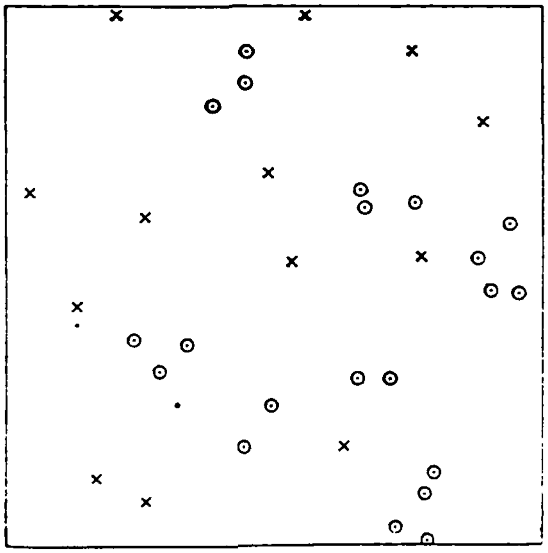


Figure 3-8: An Example of the Marked (Weighted) Point Process

$$E_{lm}(r)=k_{lm}(r)\lambda^2 g(r)dF_1 dF_2 \tag{3.5}$$

where, k_{lm}, cross-mark-correlation function, is a conditional mean of the product of the l-mark and m-mark of the members of a point-pair of distance r ; λ is intensity ; dF_1 and dF_2 are infinitesimal areas ; the radial distribution function, g(r), is related to the second-order property K by

$$g(r)=\frac{d}{dr}K(r)/2\pi r \tag{3.6}$$

The definition of mark-correlation functions k_{ll} and k_{mm} is analogous. The statistical determination of k_{lm}(r) is as follows.

1. Estimate the edge-corrected quantity $\rho_{lm}(r)$ given by

$$\rho_{lm}(r) = k_{lm}(r)\,\rho(r)$$

$$\hat{\rho}_{lm}(r) = \sum_{[x_j; l(x)]} \sum_{[y_j; m(y)]}^{y \neq x} \frac{f_\delta(|x-y|-r)\, l(x) m(y)}{2\pi |x-y|\, a(\, W_x \cap W_y\,)} \tag{3.7}$$

here, l(x) is the l-mark of x, and m(y) is the m-mark of y.

2. Compute $\hat{k}_{lm}$(r) by

$$\hat{k}_{lm}(r) = \hat{\rho}_{lm}(r) / \hat{\rho}_\delta(r) \tag{3.8}$$

here, $\hat{\rho}_\delta$(r) is the same as the edge-corrected second-moment and can be calculated using kernel estimator.

3. Transformed correlation functions are

$$g_{ll}(r) = k_{ll}(r) / \bar{l}^2,$$

$$g_{mm}(r) = k_{mm}(r) / \bar{m}^2,$$

$$g_{lm}(r) = k_{lm}(r) / \bar{l}\bar{m} \tag{3.9}$$

where, $\bar{l}$ denotes a mean l-mark and so on. These functions tend towards 1 if r goes to infinity.

Fig. 3-9 shows trace length correlations of our mapped pattern. As can be seen, this function approaches 1 as distance increases and has an approximate value 1.8 near the distance 0.2. From this result, we conclude that traces are heavily correlated with each other when they get closer, and at greater distance, they become independent.

3.4 Doubly Stochastic Point Model

3.4.1 Doubly Stochastic Point Process

In this section, we use the doubly stochastic (Cox) point process for our rock fracture model. The original idea of the Cox process (see Section 2.3.4) is as follows,

1. $\{\Lambda(x) : x \in \mathbf{R}^2\}$ is a non-negative-valued stochastic process.
2. Conditional on $\{\Lambda(x) : x \in \mathbf{R}^2\}$, the events form an inhomogeneous Poisson process with intensity function $\lambda(x)$.

The point process is stationary if and only if the intensity process $\{\ \Lambda(x)\ \}$ is stationary, and it is isotropic if and only if the intensity process is isotropic. A convenient

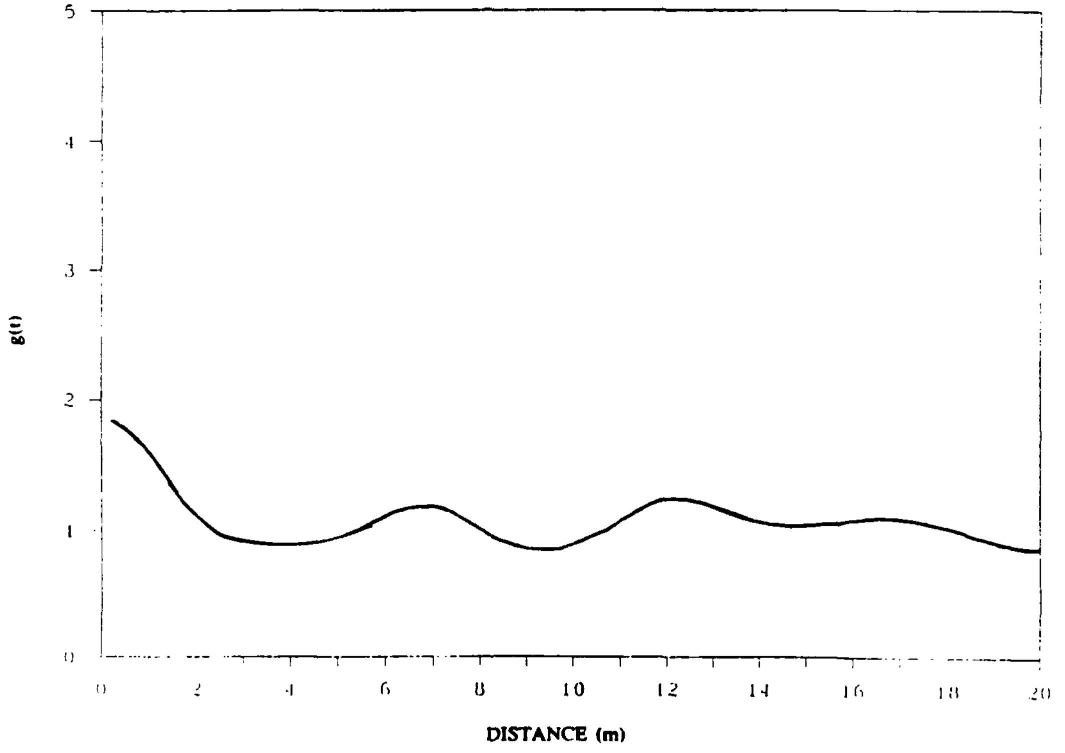


Figure 3-9: Correlations of Trace Length

and expressive terminology is to refer to the Cox process driven by { $\Lambda(x)$ }. First- and second-order properties are obtained from those of the inhomogeneous Poisson process by taking expectations with respect to { $\Lambda(x)$ }. Thus, in the stationary case, the intensity is,

$$\lambda = E[\Lambda(x)] \tag{3.10}$$

Also the conditional intensity of a pair of events at x and y, given { $\Lambda(x)$ }, is $\Lambda(x)\Lambda(y)$, so that

$$\lambda_2 = E[\Lambda(x)\Lambda(y)] \tag{3.11}$$

In the stationary, isotropic case this can be written as

$$\lambda_2(x,y) = \lambda^2 + \gamma(t) \tag{3.12}$$

where

$$\gamma(t) = Cov\{\Lambda(x), \Lambda(y)\} \tag{3.13}$$

and t is the distance between x and y, $\gamma(t)$ is the covariance density of the point process.

A simple example of the Cox process is to use the bivariate kernel function as an expansion form of the univariate kernel function which was used in Section 3.2. in the inhomogeneous point process. Assume $h(.)$ is a bivariate probability density function and define an intensity process { $\Lambda(x)$ } such that

$$\Lambda(x) = \mu \sum_{i=1}^{\infty} h(x - X_i) \tag{3.14}$$

for a non-negative value μ, where the X_i are the points of a Poisson process. One of the possible ways to construct the bivariate kernel function is to use the bivariate normal distribution function. Thus the Cox process driven by Eqn (3.14) is also a kind of Poisson cluster process. The above mentioned kernel function method can replicate the local pattern of the fracture sets, but if we need to extrapolate a regional pattern from the local mapped data, this method becomes impractical since the kernel function needs actual data sets and, in addition, simulation of the pattern is limited to a region where the data sets are available.

One of the promising ways to construct a regional pattern from the mapped data is to use the spectral density function which is frequently used in simulating the random function (see Shinozuka & Jan, 1972). We will discuss this function in the following subsection.

3.4.2 Spectral Density Function

As mentioned before, the main efforts in the doubly stochastic Poisson point process should be directed to finding the planar intensity function λ(x). In Section 3.2, we adopted the inhomogeneous point process with one-dimensional (or univariate) kernel functions and showed the corresponding modeling process. The other way to estimate the intensity function is to use the spectral density function ; this is useful if the explicit form for the intensity measures is hard to obtain.

Briefly saying, what we want to find with spectral density functions is the intensity function, λ(x,y). We assume that the intensity function can be derived from the x- and y-directional intensity measure since the mid point pattern in Fig. 3-1 shows clustering behavior in the x- and y-direction, and we also assume that each directional intensity measure is independent of the other. Then we have,

$$\lambda(x,y) = \hat{\lambda} + \sqrt{Var}\,\lambda_x(x)\,\lambda_y(y) \tag{3.15}$$

where, $\hat{\lambda}$ is a mean value of the intensity measure and, in our case, the sampling average can be used. In other words, the intensity of a region is assumed to follow a homogeneous Gaussian distribution. Since the sampling average can be obtained from data, we need to enumerate the functions λ_x(x) and λ_y(y).

To find λ_x(x) and λ_y(y), we adopt the spectral density functions. Before employing spectral density functions for each direction, we calculate some additional properties.

1. Modify the second moment in order to include the directional characteristics.
The definition of the second moment measure K(t) is

$$K(t) = \lambda^{-1} E[\, \textit{no. of further events within distance } t \textit{ of an arbitrary event}\,] \tag{3.16}$$

or equivalently,

$$\lambda K(t) = \int_0^{2\pi}\int_0^t \{\, \lambda_2(x)/\lambda \,\}\, x\,dx\,d\theta$$
$$= 2\pi\lambda^{-1}\int_0^t \lambda_2(x)\,x\,dx \qquad (3.17)$$

conversely,

$$\lambda_2(t) = \lambda^2 (2\pi t)^{-1} K'(t) \qquad (3.18)$$

Similarly, we can define a directional second moment measure as

$$K(t,\theta) = \lambda^{-1} E[\, \textit{no. of further events within distance t and within angle } \theta \textit{ of an arbitrary event} \,] \qquad (3.19)$$

or

$$\lambda_2(t,\theta) = \lambda^2 (2\theta t)^{-1} K'(t,\theta) \qquad (3.20)$$

In our case, θ is measured between the line connecting two selected points and the x-axis, and is assumed to vary within 0 to 10 deg. for the x-direction and 90 to 100 deg. for the y-direction. Fig. 3-10 shows the directional second-moment measure for the x- and y-direction, as well as the K(t) values of the mapped pattern and the homogeneous, isotropic case. From Fig. 3-10, we see the clustering behavior in the y-direction and short-range clustering behavior in the x-direction since the second moments of these two cases are greater than that of the homogeneous, isotropic case. However, as relative distance between points increases, we can see the inhibitory behavior along the x-direction. This behavior can be seen in Fig. 3-10, also.

2. Find directional covariance density for each direction. A directional covariance density function can be defined as

$$\gamma(t,\theta) = \lambda_2(t,\theta) - \lambda^2 \qquad (3.21)$$

Fig. 3-11 shows the calculated covariance densities for each direction. For comparison, a covariance density for the non-directional mapped case is also plotted.

3. Calculate correlation function ρ(t,Θ) from

$$\gamma(t,\theta) = \rho(t,\theta)\,\gamma(0) \qquad (3.22)$$

where $\gamma(0)$ is a variance of the intensity measure. If the pattern is homogeneous and isotropic, this value becomes $\gamma(0) = \lambda(1-\lambda)$.

4. For later use, derive an exponential form for the correlation function. From curve fitting, we estimate the x-directional correlation function as exp(-0.33x) and that for the y-direction as exp(-0.1y). Fig. 3-12 shows the exponential curve fitting form of the correlation function. Since t is a distance between two points (in our case, a point represents the mid point of a trace) and the correlation function ρ is an even function with respect to t,

$$\rho(t,\theta) = \rho(-t,\theta) \qquad (3.23)$$

In the general case, we assume that the n-fold Fourier transform of $\rho(t)$ exists

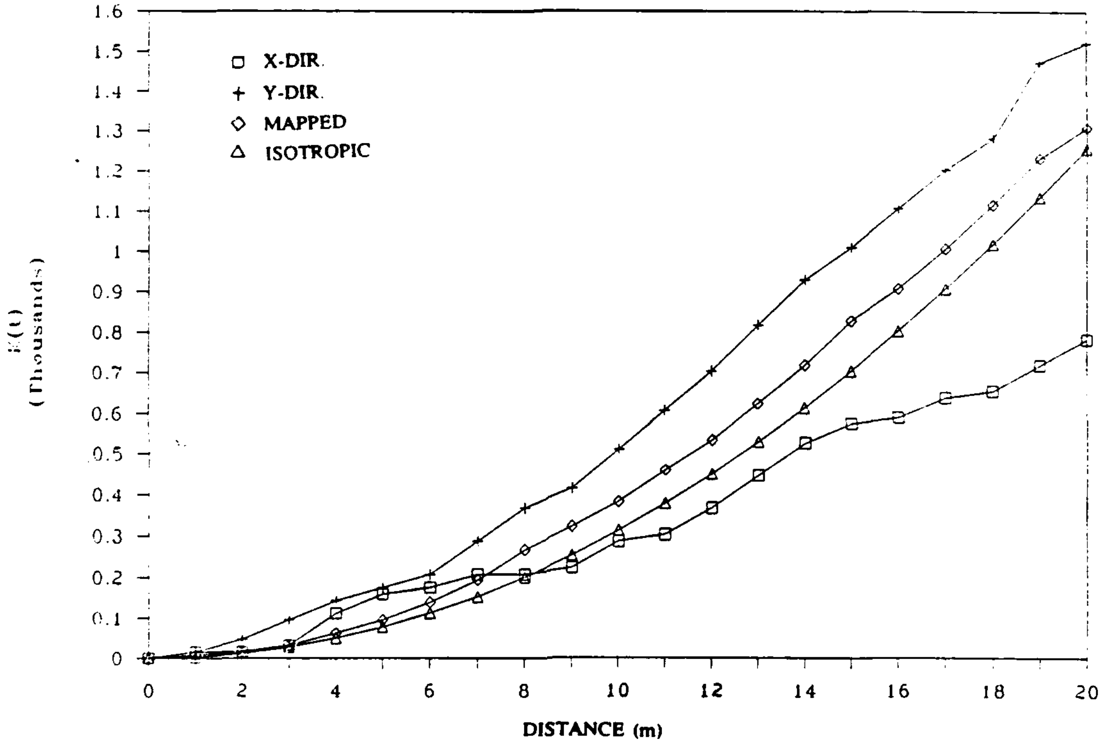


Figure 3-10: Directional Second Moment of Mapped Data

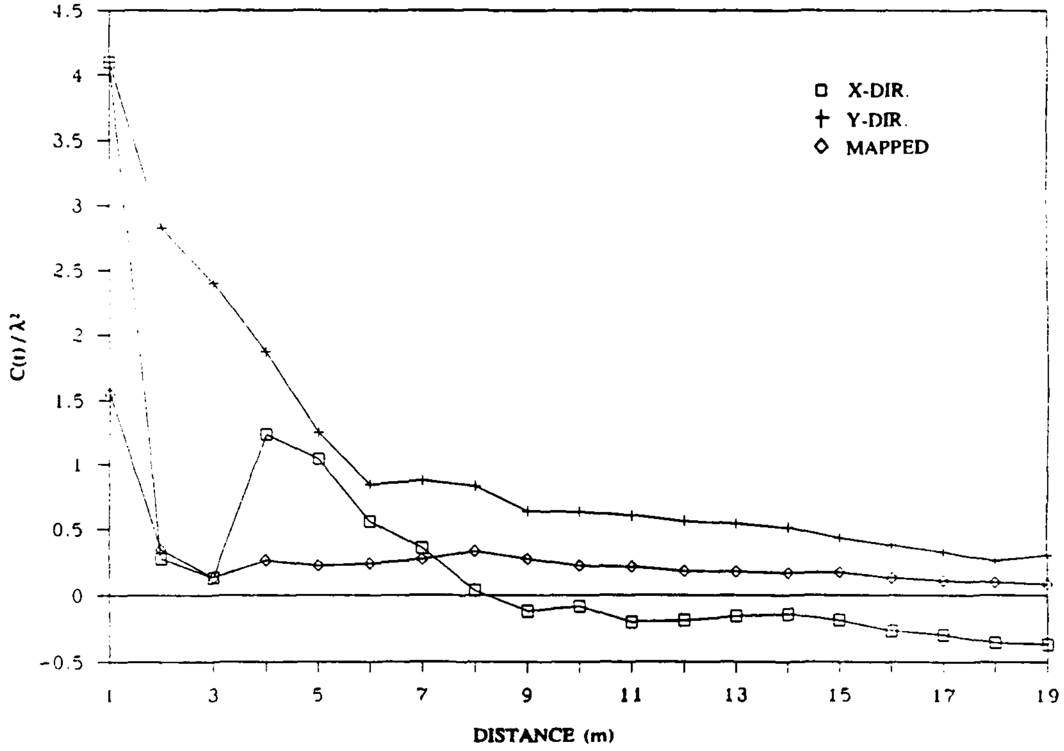


Figure 3-11: Covariance Densities for Each Case

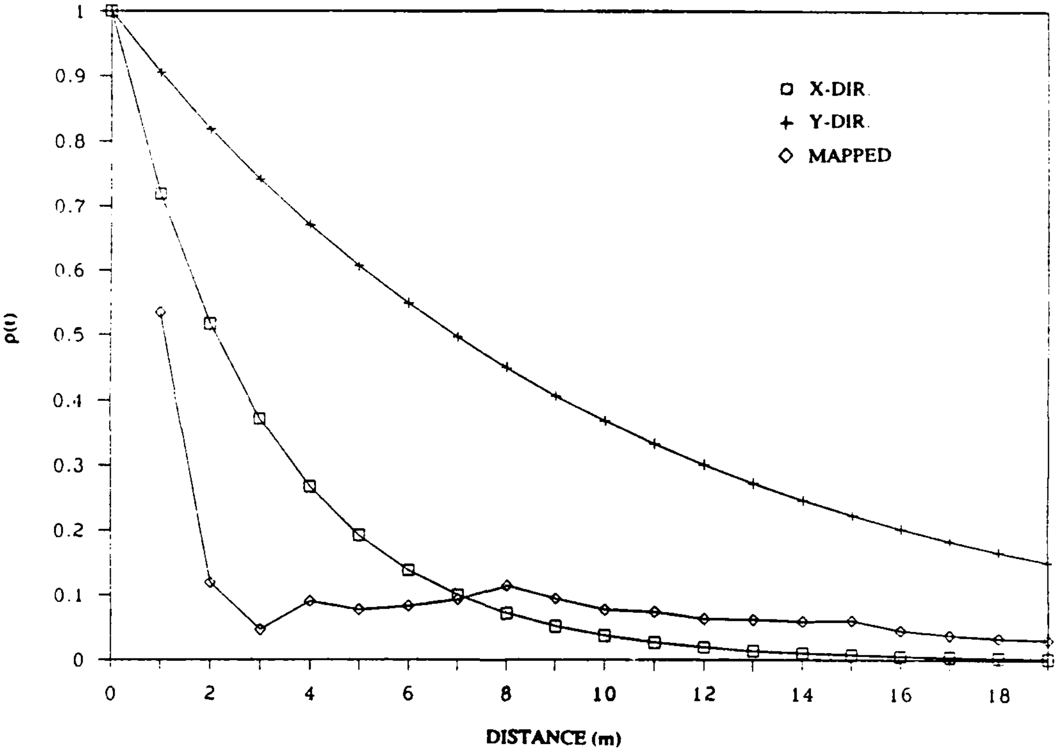


Figure 3-12: Correlation Function for X and Y Direction

(Shinozuka & Jan, 1972). The spectral density function, S(ω), of the intensity measure is then defined as

$$S(\omega)=\frac{1}{(2\pi)^n}\int_{-\infty}^{\infty}\rho(t)\,e^{-i\omega\cdot t}dt \qquad (3.24)$$

where, ω is the frequency (wave number) vector and $\omega{\cdot}t$ is the inner product of ω and t. Then from the properties of Eqn (3.23),

$$\int_{-\infty}^{\infty}\rho(t)\,sin(\omega{\cdot}t)\,dt=0 \qquad (3.25)$$

and from Eqn (3.24),

$$S(\omega)=S(-\omega) \qquad (3.26)$$

Now, if we confine our interests to a mapped pattem, n becomes 1, S(ω) becomes $S_x(\omega)$ in the x-direction and $S_y(\omega)$ in the y-direction. Then x-directional spectral density function, $S_x(\omega)$, becomes

$$S_x(\omega)=\frac{1}{2\pi}\int_{-\infty}^{\infty}\rho_x(t)\,cos(\omega{\cdot}t)\,dt \qquad (3.27)$$

and y-directional spectral density function, $S_y(\omega)$, is

$$S_y(\omega)=\frac{1}{2\pi}\int_{-\infty}^{\infty}\rho_y(t)\,cos(\omega{\cdot}t)\,dt \qquad (3.28)$$

and is real.

Since we have an exponential form of the correlation function, we can obtain a simplified form of Eqn (3.27) and Eqn (3.28). That is, if $\rho_x(t)$ = $e^{-t/a}$, then $S_x(\omega)$ = $a/\pi(1+\omega^2a^2)$. Our spectral density functions according to this form are plotted in Fig. 3-13 and Fig. 3-14.

As a result, we can estimate the x- and y-directional intensities as follows.

1. Consider the x-directional intensity function, $\lambda_x(t)$, and assume that the spectral density function becomes negligible outside the range ($-\bar{\omega}$, $\bar{\omega}$) ($\bar{\omega}$ is a cut-off frequency)

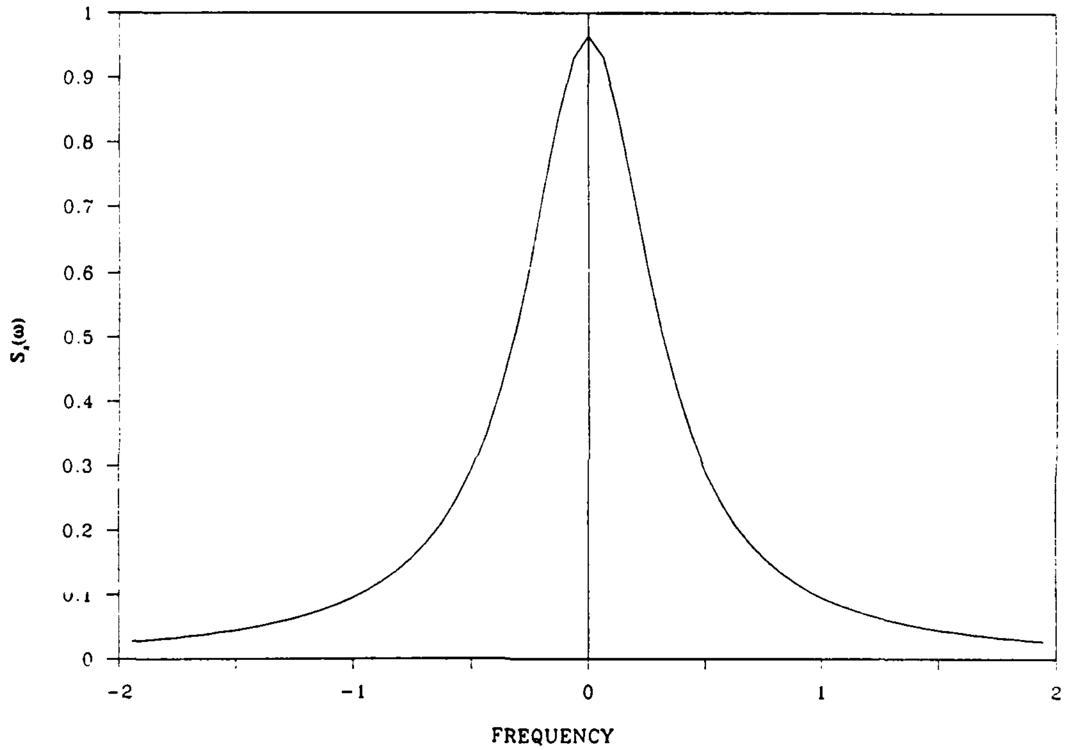


Figure 3-13: X-dir. Spectral Density Function

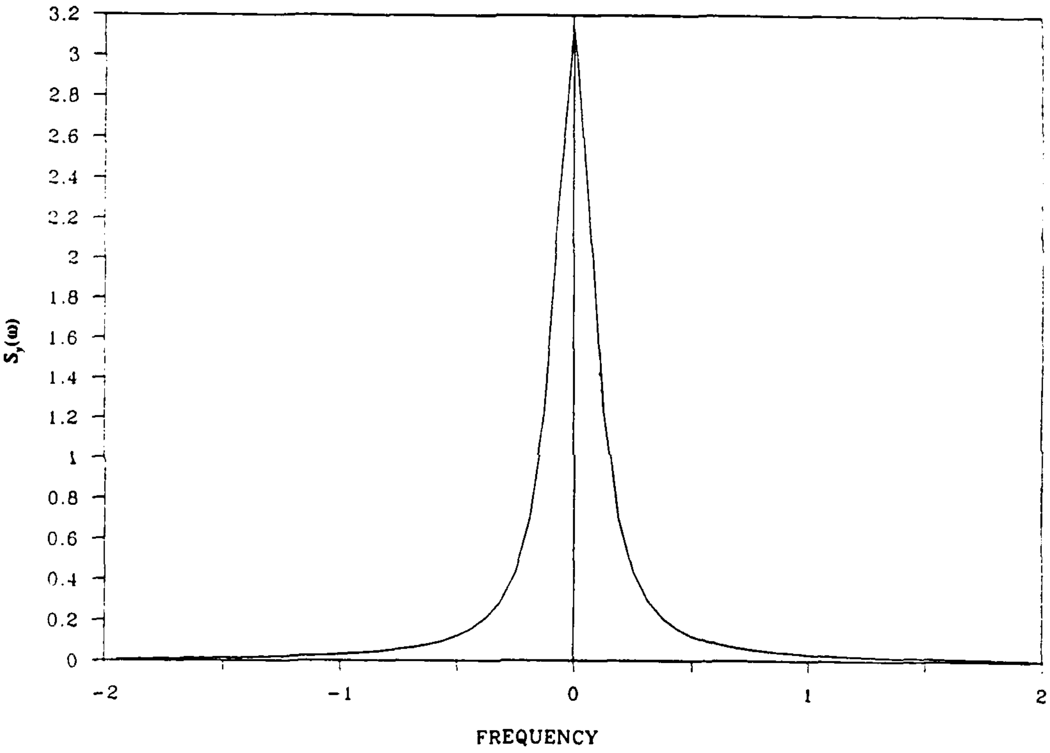


Figure 3-14: Y-dir. Spectral Density Function

2. Divide the area under the spectral density function into small areas $S_x(\omega)\Delta\omega$ at equal frequency intervals $\frac{2\overline{\omega}}{N}$.

3. Now define the x-directional intensity function, $\lambda_x(t)$, as (Augusti, et al., 1984)

$$\lambda_x(t)=\sum_{i=0}^{N}[2S_x(-\overline{\omega}+i\Delta\omega)\Delta\omega]^{1/2}cos[(-\overline{\omega}+i\Delta\omega)t+\phi_i] \qquad (3.29)$$

here, if angle ϕ_i is a realization of a random phase angle $\overline{\phi}_i$, uniformly distributed over $(0,2\pi)$, Eqn (3.29) becomes an ergodic periodic random function $\hat{\lambda}_x(t)$, with period $T=\pi/\overline{\omega}$.

Fig. 3-15 shows the schematic procedure to calculate the spectral density function.

In our case, we assume that the cut-off frequency is $\pi/2$ and the number of subdivision N is 20. The same procedure can be applied to the y-directional intensity function $\lambda_y(t)$. With the aforementioned procedure, we can find the intensity function, Eqn (3.15), in a region which is a realization of the Cox process.

Simulation of the Cox process driven by the spectral density functions can again be done with the Acceptance / Rejection scheme which was discussed in Section 3.2. But in this case, the A / R process should be used twice for the x- and y-direction (two-phase A/R scheme) because the directional intensity function derived from the spectral density function may have negative values. One of the simulated patterns is shown in Fig. 3-16 where we can see the clustering behavior of the simulated point pattern which is similar to that in the original pattern (see Fig. 3-1). Again, the second-moment analysis as well as the Monte-Carlo test show that the Cox model fits our mapped pattern (Fig. 3-17).

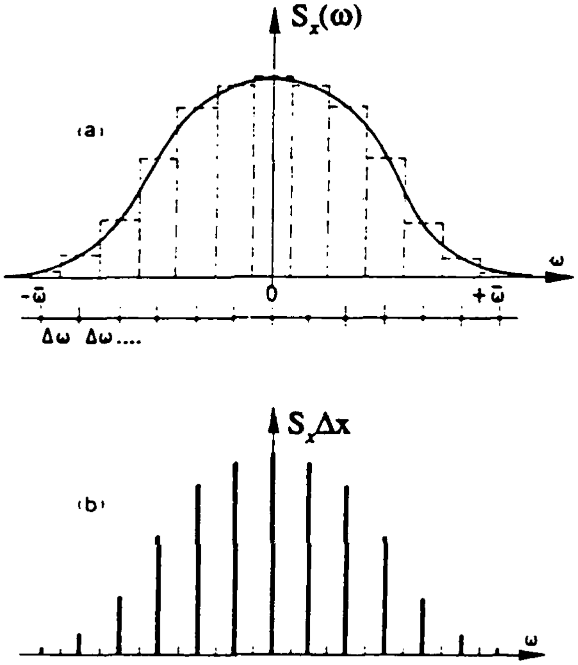


Figure 3-15: Calculation of Spectral Density Function

3.5 Conclusion

We conclude from the above results that :

1. Our mapped pattern (Fig. 2-1) does not follow a homogeneous Poisson point pattern even though our case has a rather simple fracture pattern.

2. An inhomogeneous Poisson point process driven by a :kernel function method is proposed as a model for our mapped pattern. However, the kernel functions summarize all the variations on the x-axis and, therefore, intensity variations in the y-direction cannot be expressed.

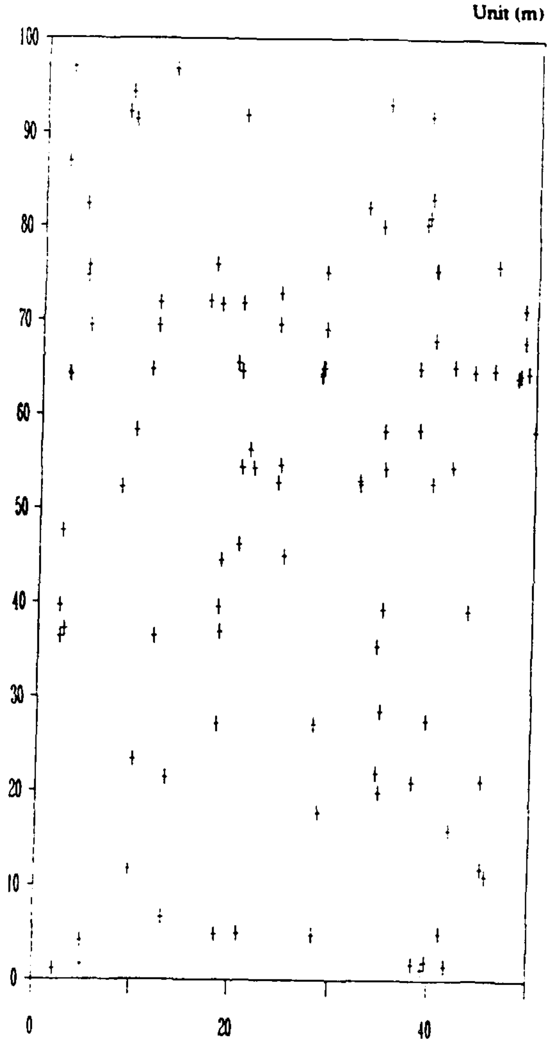


Figure 3-16: Mid Point Pattern Using Cox Process

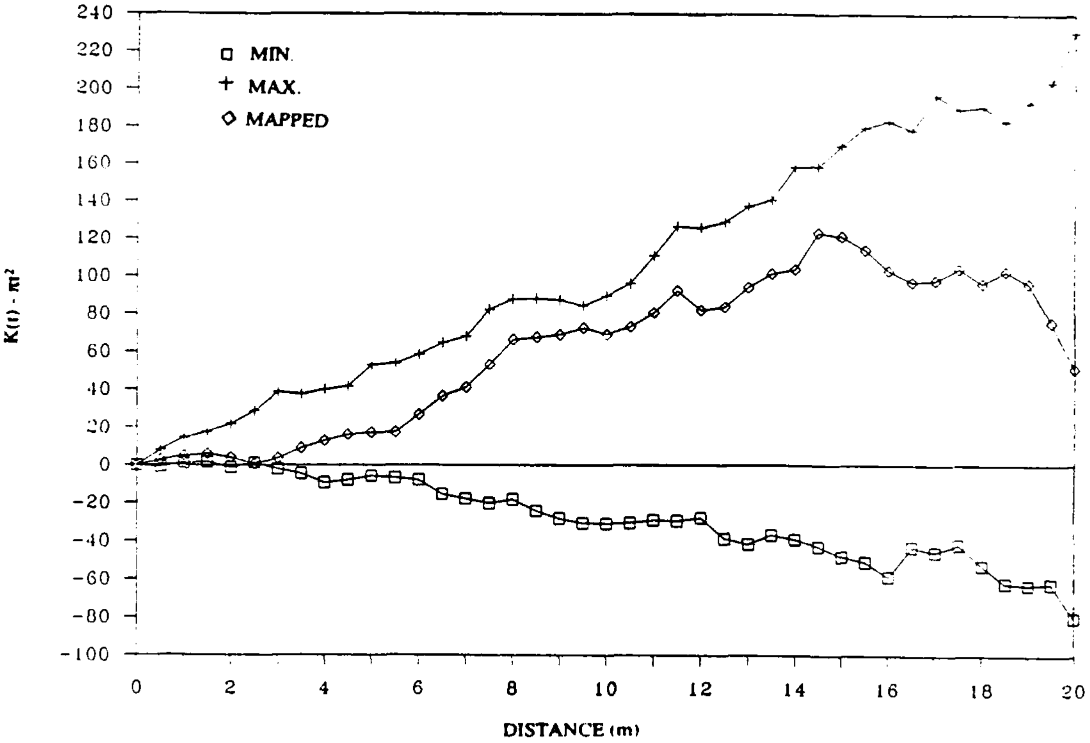


Figure 3-17: Second Moment Analysis of Cox Process

3. Nevertheless, using a kernel function with normal distributions, shows the regional inhibitory behavior in the x-direction. In the general case, one could further expand the kernel function to two-dimensional bivariate kernel functions ; this is not done because the Cox process (see 5 below) provides a better approach.

4. Trace lengths are correlated as they get closer. That is to say, a clustering mechanism among traces can be predicted.

5. With the doubly stochastic (Cox) process, we are using a directional second-moment measure and find the directional correlation function. Also, we adopt a spectral density function with which we are able to transform the random function into the frequency domain. From the assumption of the stationary Gaussian random distribution, we derive the intensity distribution and perform simulations using a two-phase A / R process. The resulting doubly stochastic point pattern also shows a directional distribution. The doubly stochastic point process model fits the mapped pattern with satisfactory goodness of fit both using the second-moment analysis and using the Monte-Carlo test.

6. As just stated, using a doubly stochastic process, we transform the intensity measure into frequency domain and derive a spectral density function. This means that, once we have a spectral density function for a particular area, we can directly reproduce a similar pattern in a larger area. This is an important advantage of the Cox process compared to the inhomogeneous Poisson point process.

7. From the point of view of rock fracture modelling, we prefer the doubly stochastic point process model to the inhomogeneous Poisson point process model. This is so because the doubly stochastic point process model is based on the correlation measures of the mapped data while the inhomogeneous Poisson point process model is not.

Chapter 4

FIBER (or LINE - SEGMENT) PROCESSES

4.1 Introduction

As described in Chapter 1, the fiber process is required for fracture geometry modelling if the traces have complex inter-actions such as intersections. In this chapter, we only consider, for the time being, two dimensional cases. Applications of the fiber processes to two-dimensional planar rock surfaces can be deduced from the three-dimensional Baecher model (Baecher, et al., 1977) or can be found in Long, et al. (1982). However, these models assume that the mid point of each trace follows a homogeneous Poisson point process with a lognormal or an exponential trace length distribution and with a uniform (or uni-directional) orientation distribution. In some cases, these assumptions are not appropriate and some modified models have been developed. In the following, we briefly introduce these recent developments in modeling of rock fractures in two dimensions and discuss their feasibility. Also, a new rock fracture modeling scheme which accounts for sequential generation of fractures is proposed and a new method to evaluate the trace length distribution function is considered.

4.2 Recent Trend in Fiber Modeling

The following developments of two dimensional fracture configuration modeling have taken place in the recent past :

4.2.1 Homogeneous Poisson Fiber Process (Baecher) Model

This model is identical to that of the homogeneous Poisson point process model except that it includes fiber length distribution and orientation distribution functions. Assuming more than one fracture set, each set of fractures is generated independently using a homogeneous, isotropic Poisson point process. Then the individual sets are superimposed. The location of each fracture in a set is designated by assuming that the centers of the fractures are randomly distributed (Poisson type) within the generation region. For each set, a density is supplied to determine the total number of fracture centers to be generated. The orientation of each fracture in a set is given next. The length of each fracture is chosen independently. Fracture lengths within a set are assumed to be distributed lognormally or exponentially. As a mark or weight, apertures can be assigned to each fracture with assumed distributions (e.g., lognormal distribution). Fig. 4-1 shows schematic procedures for constructing the homogeneous Poisson fiber process.

Some mathematical characteristics for the homogeneous Poisson fiber process can be found for each fracture set (see, Parker & Cowan, 1976, Cowan, 1979 and Stoyan, et al., 1987) ;

1. Intensity of fibers (L_A) : The total length $\Phi(w)$ of all fiber pieces in a plane W can be measured easily if an image-analyzer is employed, or if the fibers are all segments or straight lines. An unbiased estimator for L_A is given by

$$\hat{L}_A = \Phi(w)/v_2(w) \tag{4.1}$$

where, v_2 is a Lebesgue measure on R^2. Otherwise, intersections with sampling lines or circles can be used.

2. Second-order Characteristics (K(t)) : Define $L_A K(t,\alpha)$ analogous to that of a point process. $L_A K(t,\alpha)$ is the mean total fiber length in a sector with radius t and angle α

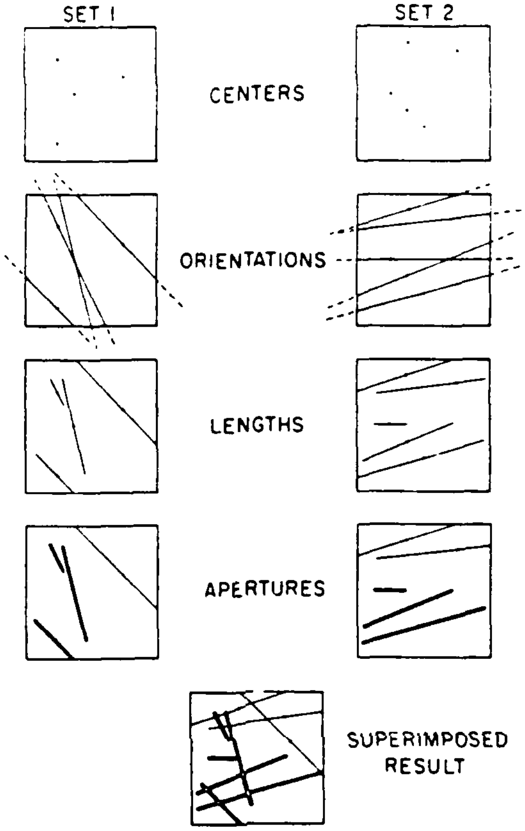


Figure 4-1: Schematic Procedures for Constructing the Homogeneous Poisson Fiber Process Model (After Long, et al., 1982)

centered at a typical fiber point. If the fiber process is motion-invariant, the typical fiber point can be explained by Palm distribution (Stoyan, et al., 1987). The second-order characteristics of the motion-invariant Poisson fiber processes are defined as

$$K(t) = L_A E[\,\textit{total lengths of further fibers within distance } t \textit{ of a typical fiber point}\,] \tag{4.2}$$

and, especially,

- If we consider the Poisson infinite line process with assumed fiber intensity, L_A,

$$K(r) = \pi r^2 + 2r/L_A, \quad r \geq 0 \tag{4.3}$$

This sum has two main components ; $2r/L_A$ is the contribution from the infinite line containing the typical point and πr^2 is formed from the remainder of the process.

- If we consider the Poisson constant unit segment process,

$$\begin{aligned} K(r) &= \pi r^2 + 2r(1-\frac{r}{2})/L_A \quad if \;\; r \le 1 \\ &= \pi r^2 + 1/L_A \qquad\qquad if \;\; r > 1 \end{aligned} \tag{4.4}$$

- If we consider the Poisson segment process where segments have random lengths,

$$L_A K(r) = L_A \pi r^2 + \int_0^r x dF_0(x) + \int_r^\infty (2r - \frac{r^2}{x^2}) dF_0(x), \quad r \ge 0 \tag{4.5}$$

where

$$\begin{aligned} &F_0(x) = \int_0^z z\, dF(z)/m, \quad x \ge 0 \\ &F(z)\,;\ \textit{length distribution function} \\ &m \;\ ;\ \textit{mean length} \end{aligned} \tag{4.6}$$

Even though the above cases have explicit forms, applications of these characteristics to mapped data are not easy since, from a practical point of view, the typical fiber point lies at random on a fiber. Hence to test the homogeneity of given mapped data is almost impossible. Here, the meaning of the typical fiber point is different from that of the mid point in that the typical point is an arbitrary point in a fiber and is only used when calculating the second-order characteristics, whereas the mid point is a fixed point which becomes a center of a fracture. Stoyan, et al. (1987) suggested approximate test schemes by introducing a test system of parallel lines or of a Poisson line process. However, this scan line method is also impractical because the spacing of the test system or the fiber intensity of the Poisson line process influence the second-order characteristics of mapped data. This particular property of the fiber processes makes a homogeneous Poisson fiber process model hard to check against mapped data.

In rock fracture modeling, Some modifications appeared after the original homogeneous Poisson fiber (two-dimensional Baecher) model. Dershowitz (1988) showed an Enhanced Baecher Model by using a termination probability. If the traces intersect the pre-existing traces, these latter traces are terminated at the intersection point with given probability. The termination probability is a ratio of the termination points and the total number of traces in a map. Dershowitz also suggested a non-circular fracture shape such as polygons and ellipses. Fig. 4-2 shows an Enhanced Baecher model with non-circular fracture shape.

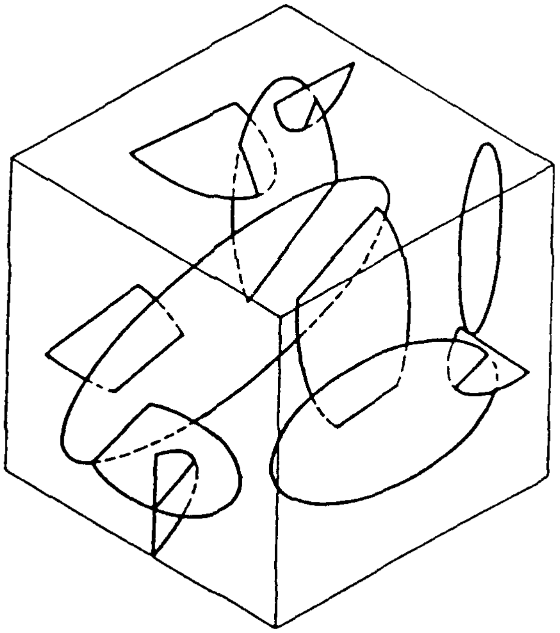


Figure 4-2: Enhanced Baecher Model

4.2.2 Parent - Daughter Model (Witherspoon and Long, 1987)

Witherspoon & Long (1987) chose a Poisson cluster process (see Section 2.3.3.) when they modeled fractures in swarms or zones. They worked in three dimensional space assuming disk-shaped fractures, but a two dimensional parent-daughter model can be deduced from their work. The location of the parents may be purely random, i.e., a fixed rate Poisson process, or it may be a regional variation in the density of the parents. Once the parents have been determined, the daughters are found at some distances from the parents with certain distribution (e.g., Gaussian). Fig. 4-3 is a three dimensional parent-daughter model. The implementation of the model requires knowledge of the distribution of the daughters per parent and the daughters around the parents as well as the density of the parents.

There are, however, certain problems when using this model. First, several swarms may overlap. Thus, it will be uncertain as to which parent a trace daughter belongs. Hence, dividing the overall density into that of parents and daughters becomes ambiguous. Second, the parent - daughter model, in most cases, shows clustering behavior near the parents and inhibitory behavior at greater distances from the parents according to a specified distribution form of daughters. Thus a fitting procedure for a model is not easy to apply.

4.2.3 Fractal Model

The usual method when considering fractal geometry is to use the fractal dimension as a main tool. If the given data follow linear relations between the number of data and the sampling size in log-log plot, the fracture network is a self-similar fractal and the fractal dimension is the value of slope in log-log plot whether the data are the trace lengths (see Barton & Larson, 1985) or the polygons made by trace intersections (La Pointe, 1988). But, the fractal dimension itself does not give us any information on the model and there is little progress in modeling rock fractures using fractal geometry.

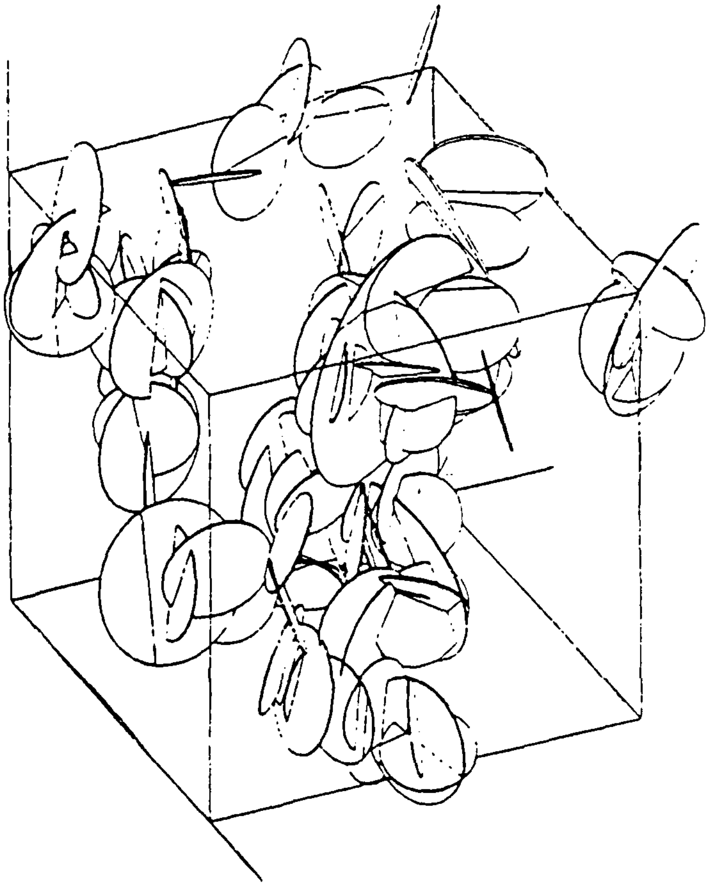


Figure 4-3: Parent-Daughter Model in Space, after Chiles (1988)

Chiles (1988) suggested one- and two-dimensional simulation methods with fractal geometry. But, as he mentioned, it is not clear whether the simulation procedure is related to the fractal dimension or not.

4.2.4 Branching Model (Watanabe, 1986)

This model is probabilistic rather than stochastic as far as generating the branches is concerned. The main idea is to calculate the branching probabilities at each branching point and check the results against the total existing length. As seen in Fig. 4-4, each fracture is divided into many branches of constant length L_0. Then, step numbers are assigned at each branch. Using branching probabilities P_n, clustered fractures are generated.

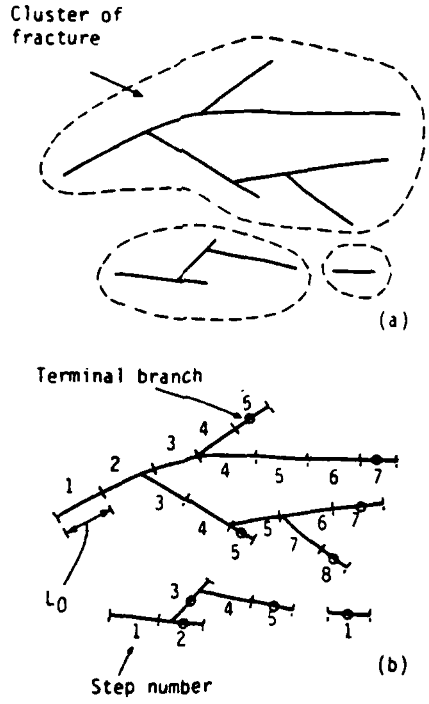


Figure 4-4: Branching Model of Watanabe (1986)

4.2.5 Hierarchical Model (Conrad & Jacquin, 1975)

Conrad & Jacquin (1975) used the terminology 'hierarchical' when they simulate fractures by using Poisson-type polygons and Boolean-type fissures (see also Serra, 1980). Their hierarchical model which was applied in rock fracture modeling is used to explain the sequential generation of a Boolean model in a domain. The concept is as follows.

1. The major network, which we name the first set, is a network of Poisson straight lines (or the realization of the Poisson line process) of density $\lambda_1(d\alpha)$ determining convex polygons : the matrix blocks. The infinite lines in Fig. 4-5 represent the matrix blocks.
2. Intersect the interior of each polygon with a Boolean diagram of segments with density λ_2 and distribution function $F(dl, d\alpha)$, i.e., in each polygon we have a realization of a single Boolean diagram of segments in which their lengths are extended to the perimeter of the polygon. In two distinct polygons the Boolean diagrams are independent and have the same characteristics. The finite segments in Fig. 4-5 are the realization of the Boolean diagram of segments.
3. The matrix block generated by the above two steps is defined by the first set, but the second set is important when describing the minor network.
4. The parameters introduced in this model are
   - density $\lambda_1(d\alpha)$ of the Poisson line process
   - density of the Poisson segment process λ_2
   - distribution function of the primary grain $F(dl, d\alpha)$ which represent the length and orientation of segments.

Fig 4-6 shows a simulated rock fractures network using the hierarchical model of Conrad & Jacquin (1975). This model differs from the Veneziano model (1979) in that the former considers polygons made by Poisson line processes as rock blocks and remaining fibers made by a Boolean diagram of segments as rock fractures, whereas the latter one, according to the persistence parameters, chooses some of the polygons made by Poisson plane processes and Poisson line processes as rock blocks.

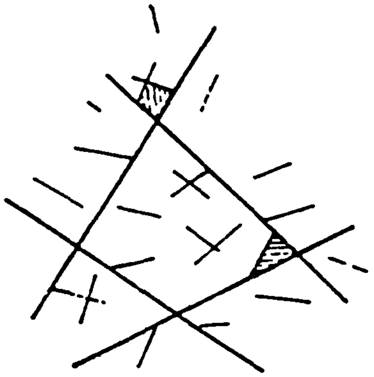


Figure 4-5: Hierarchical model of Conrad & Jacquin (1975)

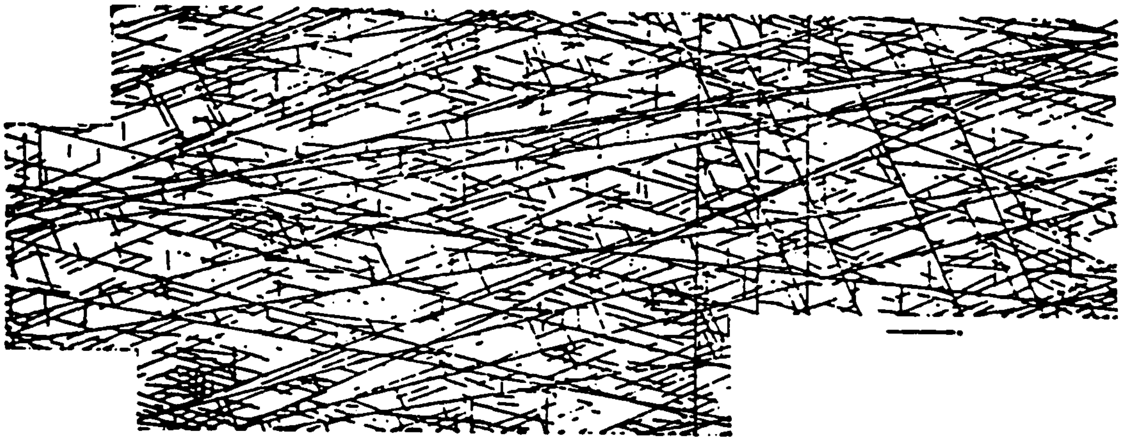


Figure 4-6: Simulated Rock Fractures Network, After Conrad Jacquin (1975)

4.2.6 Percolation Model

In percolation theory, a medium is considered as an infinite set of sites and a bond is assumed to be a path which connect certain pairs of sites. Therefore, two different approaches are possible when modeling fracture systems with percolation theory. The one is site percolation theory by Robinson (1983). In this model, segments of specified length and orientation distributions are uniformly generated in a square. By way of the trial and error method, one can find the critical value of segment intensity and length scale with which percolation is to occur. Thus we can idealize fracture systems in a plane.

The other is bond percolation by Watanabe (1986). Assume an original network composed of many sites linked together by bonds. The network is essentially defined by the number of bonds z initiating from each site. Next, by taking out some of the bonds from the original network, the modified networks can be constructed. In this course of modification, the decision as to whether each bond may remain or be removed is independently established using a fixeď existence probability P_b, where P_b can be calculated from a map or can be assumed. If P_b is large, many bonds remain and some large clusters linked with each other will remain.

4.2.7 Crack Tessellation Model (Gray, et al., 1976)

One of the interesting models of random tessellation is to adopt the crack model. With some modification, a non-homogeneous Poisson model can be developed. A marked point process Φ is constructed on the plane, each point of Φ being marked with the orientation of a line. The edges are produced by a growth process : each edge starts at one of the points and grows in the two opposing directions specified by the marks at a constant rate. Growth in a particular direction continues until further edges are hit. This process is reminiscent of crack growth.

4.3 Proposed Hierarchical Model

4.3.1 Background

The original hierarchical model which was applied in rock fracture modeling (Conrad & Jacquin, 1975) was used to explain the sequential generation of a Boolean model in a domain. We are now trying to modify the idea of the original hierarchical model and apply it to the rock fractures, where the rock fracture is assumed to be generated by geological sequences. When we deal with the modified model, we assume that the fracture generation sequence is known apriori. We can, therefore, model the hierarchical fracture sequence as follows.

1. We divide the fractures into several groups according to the geological history. For simplicity, we assume that there are two sets such as set 1 and set 2.
2. From the assumed or calculated intensity, orientation and trace length distribution, generate set 1 of fractures on the basis of the point process.
3. Do an independence test between set 1 and the set 2. For this, we replace a trace by a point (i.e., mid point) and introduce the bivariate point processes. If we find that these two sets have no dependent relations, ignore step 4.
4. Before generating set 2, measure the interrelations (i.e., correlations) between two sets, where each set of traces is equivalent to a set of points.
5. If there are no such correlations, set 2 is obviously independent of set 1, and the superimposition of set 2 is possible. If we find the correlations between two sets, the A / R procedure can again be used to simulate set 2. If we want to impose a termination of set 2 when it crosses set 1, we can define a termination condition at this stage.
6. Iterate step 2 to step 5 according to the number of fracture sets.

In the following, we will discuss our proposed hierarchical model in detail.

4.3.2 Independence Test

Assume that the process consists of two sets of traces classified as type 1 events and type 2 events by their geological history. If the mid points of each set are distributed independently of each other, we can superimpose traces set by set. But if there are correlations between the two sets, we need to include these correlations into our hierarchical model. For this, we use the bivariate point process when performing the independence test.

Extension of the univariate point process (Chapter 2) to the bivariate case is relatively straightforward and will now be discussed. Define a bivariate point process which generates events classified as type j for $j = 1, 2$. First-order properties are determined by each intensity.

$$\lambda_j = \lim_{|d\mathrm{x}| \to 0} \left(\frac{E[N_j(d\mathrm{x})]}{|d\mathrm{x}|} \right) \tag{4.7}$$

and second-order properties are

$$\lambda_{ij}(u) = \lim_{|d\mathrm{x}||dy| \to 0} \left(\frac{E[N_i(d\mathrm{x})]\,E[N_j(dy)]}{|d\mathrm{x}||dy|} \right) \tag{4.8}$$

where the infinitesimal regions dx and dy are centered on points x and y a distance u apart. Stationarity and isotropy imply that $\lambda_{ij}(u) = \lambda_{ji}(u)$.

The bivariate extension of second moment $K(t)$ can be defined as

$$K_{12}(t) = \lambda_2^{-1} E[\textit{number of type 2 events within distance } t \textit{ of an arbitrary type 1 event}] \tag{4.9}$$

To relate the $K_{12}(t)$ to the corresponding $\lambda_{12}(u)$, note that the conditional intensity of a type 2 event at a specified location a distance u from the origin, given a type 1 event at the origin, is $\lambda_{12}(u) / \lambda_1$. Like in the univariate case,

$$K_{12}(t) = 2\pi(\lambda_1\lambda_2)^{-1}\int_0^t \lambda_{12}(u)\,u\,du \tag{4.10}$$

It follows that $K_{12}(t) = K_{21}(t)$.

Cross-covariance densities are defined as

$$\gamma_{12}(u) = \lambda_{12}(u) - \lambda_1 \lambda_2 \tag{4.11}$$

In practice, if we assume two independent components, type 1 and type 2, the dual intensity of a type 1 event and a type 2 event at any two specified locations is $\lambda_1\lambda_2$. In other words, the second-order intensity function $\lambda_{12}(u)$ assumes a constant value $\lambda_1\lambda_2$ for all distance measure u, then Eqn (4.10) gives

$$K_{12}(t)=\pi t^2 \tag{4.12}$$

and Eqn (4.11) becomes zero.

Here, independence is analogous to complete spatial randomness for a univariate pattern in that it provides a convenient reference point for the characterization of the more interesting bivariate structure. Generally speaking, we say that components 1 and 2 are positively (negatively) correlated at range t, if the derivative of $K_{12} - \pi t^2$ with respect to t is positive (negative) (Diggle, 1983).

4.3.3 Correlation Measure

As mentioned before, if two sets of traces are not independent, correlation measures between sequential steps become important since two sets of traces are inter-correlated and, thus, there exist correlations which need to be included when using the hierarchical model. Here, we will discuss two basic ideas which can be used when calculating the correlations.

We can generate the hierarchical sequences of the two fracture sets by conditioning the mid points pattern of set 2 on the traces of set 1. This is done by using the line-kernel function method for representing traces of set 1 and by using mid points for representing set 2. For use of a line-kernel function, we divide each trace into a set of segments in which all segments have the same length. For each segment, the center point is obtained. Then we can include the length effect of a fiber process by evaluating the line-kernel function over all center points of set 1. The resulting line-kernel function can be visualized as a contour plot around the fiber which has a peak value at the mid point of the fiber. This method requires complicated numerical work when calculating the distances between the mid points of set 2 and the evaluated center points of set 1. From the measurement of the distances, we have

$$\begin{aligned}\hat{\lambda}(t|T_1) &\approx \sum \int_{T_{1_i}} h(t,l)\,dL(l) \\ &= C_1(\sigma)[F(t|\sigma)+C_2]\end{aligned} \tag{4.13}$$

where, $h(t,l)$ is a line-kernel function with segment length l and parameter σ ; $C_1(\sigma)$ and C_2 are the coefficients of the line-kernel function method. Once we have the conditional distributions for the mid points of set 2 with respect to the fiber lengths of set 1, we can also include the orientation parameter in the line-kernel function.

The other way to measure the correlations is to modify the nearest neighbor distance method of Chapter 2. Here one uses the nearest fiber distance from the mid points of set 2. Any position of a fiber can be chosen when calculating the nearest neighbor distance, and this flexible position of the nearest fiber method makes it possible to include the fiber length effect in a correlation measure. Measuring the nearest neighbor distance, we have

$$\hat{\lambda}(t|T_1)=C_3(\sigma)[G(t|\sigma)+C_4] \tag{4.14}$$

where, $G(t|\sigma)$ is an assumed nearest neighbor distance function with parameter σ ; $C_3(\sigma)$ and C_4 are the coefficients of the nearest neighbor method. The parameters and coefficients of these two methods can be found if we use MLE for a given data set.

Stoyan & Ohser (1984) proposed another method to calculate the correlations between two sets of fibers and developed a mathematical correlation formula for the fiber processes which are based on the second-order characteristics. But their method uses a scan line method when choosing a typical point and also requires measurement of the intersection angle between a scan line and a fiber. Thus, estimation of the correlations is further complicated.

4.4 Trace Length Distribution Function

Before we discuss the application of the hierarchical model, we need to develop a trace length distribution. In general one knows that three sampling biases for trace lengths exist when we adopt a scan line method (Einstein and Baecher, 1983).

- Proportional Length Bias : Longer trace lengths have a proportionally larger probability of intersecting the line and therefore of being sampled.
- Censoring Bias : Many of the traces observed in the excavation run off into the rock walls or the overburden, and cannot be observed in their entirety. Since this censoring occurs with proportionally higher probability to longer traces, the sample is biased toward shorter lengths and the extreme upper tail disappears completely. If we assume exponential distributions, we have a closed form for the mean trace length using maximum likelihood.

$$\hat{l}_{ML} = \frac{r}{\sum l_{x,i} + \sum l_{z,j}} \tag{4.15}$$

where, $l_{x,i}$ is a set of completely observable traces (i = 1,...,r) and $l_{z,j}$ is a set of traces for which only one or neither end is observable (j = 1,...,t).

- Truncation Bias : Traces shorter than some cut-off length are not recorded.

As a modification of this existing bias correction, we developed a new approach to correct the first two biases :

The following Maximum Likelihood Estimators (MLE) can be used where a trace is partitioned into three groups.

- $u = \{ u_1,...,u_a \}$: Trace lengths with both ends observable
- $v = \{ v_1,...,v_b \}$: Trace lengths with one end observable
- $w = \{ w_1,...,w_c \}$: Trace lengths with no ends observable

Now our MLE for the density function of trace length distribution $f_x(x)$ becomes

$$l(\{u_i\},\{v_j\},\{w_k\}|f_x(x)) = \prod_{i=1}^{a} P[u|L_i]f_{u|L_i}(u_i)\prod_{j=1}^{b} P[v|L_j]f_{v|L_j}(v_j)\prod_{k=1}^{c} P[w|L_k]f_{w|L_k}(w_k) \tag{4.16}$$

Here,

$$\begin{aligned} P[u|L] &\ \alpha\ \int_0^L f_x(x)(L-x)\,dx \\ P[v|L] &\ \alpha\ 2\int_0^L f_x(x)x\,dx + 2\int_L^\infty f_x(x)L\,dx \\ &= 2\int_0^\infty \min(x,L) f_x(x)\,dx \\ P[w|L] &\ \alpha\ \int_L^\infty f_x(x)(x-L)\,dx \end{aligned} \tag{4.17}$$

and

$$\begin{aligned} f_{u|L}(u_i) &\ \alpha\ (L-u_i)f_x(u_i) \\ f_{v|L}(v_j) &\ \alpha\ 1-F_x(v_j) \\ f_{w|L}(w'_k) &\ \alpha\ \ 1 \end{aligned} \tag{4.18}$$

here, L is a window length of the map. Fig 4-7 shows schematic estimation of our MLE.

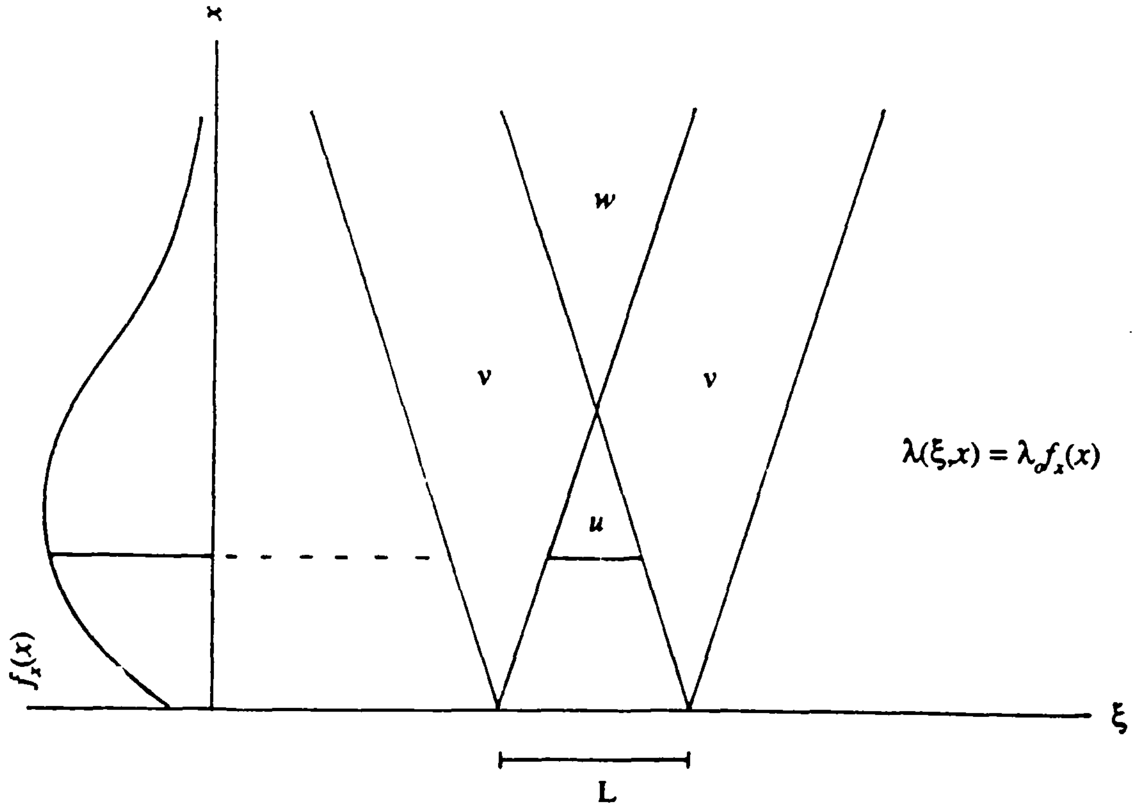


Figure 4-7: Schematic Estimation of Trace length

4.5 Discussion

In this Chapter, we have shown various stochastic fracture models including a homogeneous Poisson fiber process and proposed our hierarchical model in which geological history of the fracture genesis can be represented. As was predicted, the hierarchical model needs the fiber density function as well as length and orientation distribution functions for the sequential sets. We, therefore, have discussed the possible measurement statistics of the independence test and correlation measurements in order to find the fiber density functions. The main difference between this modified hierarchical model and the existing models is the realization of the correlated behavior of the sequential sets which is not fully implemented in the existing models. In addition, we developed a new technique to measure the mean trace length by utilizing MLE.

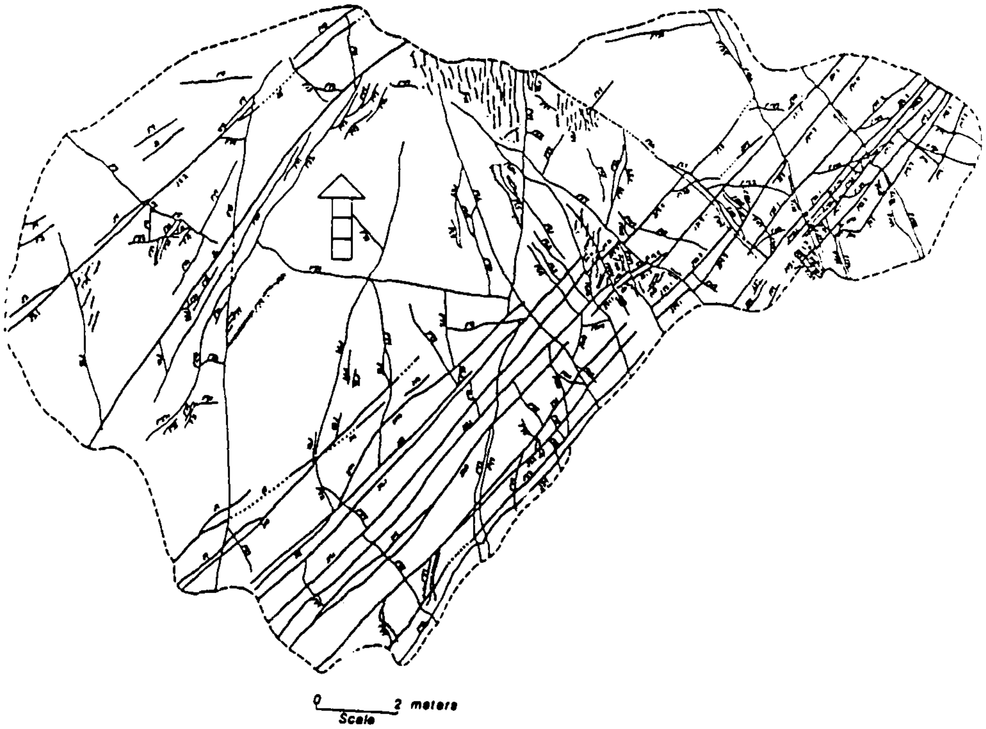
In the next Chapter, we will apply the above mentioned model and related statistics to real mapped data and discuss the feasibility of our model.

Chapter 5

APPLICATION OF THE HIERARCHICAL MODEL

5.1 Introduction

In applying the point process for modeling of fracture patterns, we have chosen a simple mapped pattern (Fig. 2-1) and developed appropriate fracture models based on the doubly stochastic point process and based on the inhomogeneous Poisson point process (see Chapter 3 for details). On the other hand, if the mapped pattern involves complex interactions among traces or fracture termination points, the point process is not an appropriate method for modeling fracture patterns. A further step is therefore to introduce a fiber process model, and to incorporate it in a hierarchical model. Fig. 5-1 is chosen as the mapped pattern (see Barton & Larson, 1985) to be modeled. It shows intensely fractured zones and complex interrelations among traces. Fig. 5-2 is a digitized trace map of Fig. 5-1 without the intensely fractured zones (shaded in Fig. 5-1). For the use of the hierarchical model, we divide the pattern in Figs. 5-1 and 5-2 into two characteristic sets. Set 1 is the group of fractures in which all traces have uni-directional orientations and have no common intersection points with each other (see, Fig. 5-3), while set 2 represents the remaining traces (see, Fig. 5-4) excluding, however, the intensely fractured zones (shaded in Fig. 5-1). Incidentally, our assumption for the two sub-sets is identical to the classification used by Barton and Larson (1985) who grouped the traces according to their apparent aperture width. The mid point patterns of the traces shown in Fig. 5-2 to Fig. 5-4 are shown in Fig. 5-5 to Fig. 5-7, respectively, assuming that the half length point is identical to the mid point.



Pavement 100 (area = $214m^2$)

Figure 5-1: Trace Map of PA100 (From Barton & Larson, 1985)

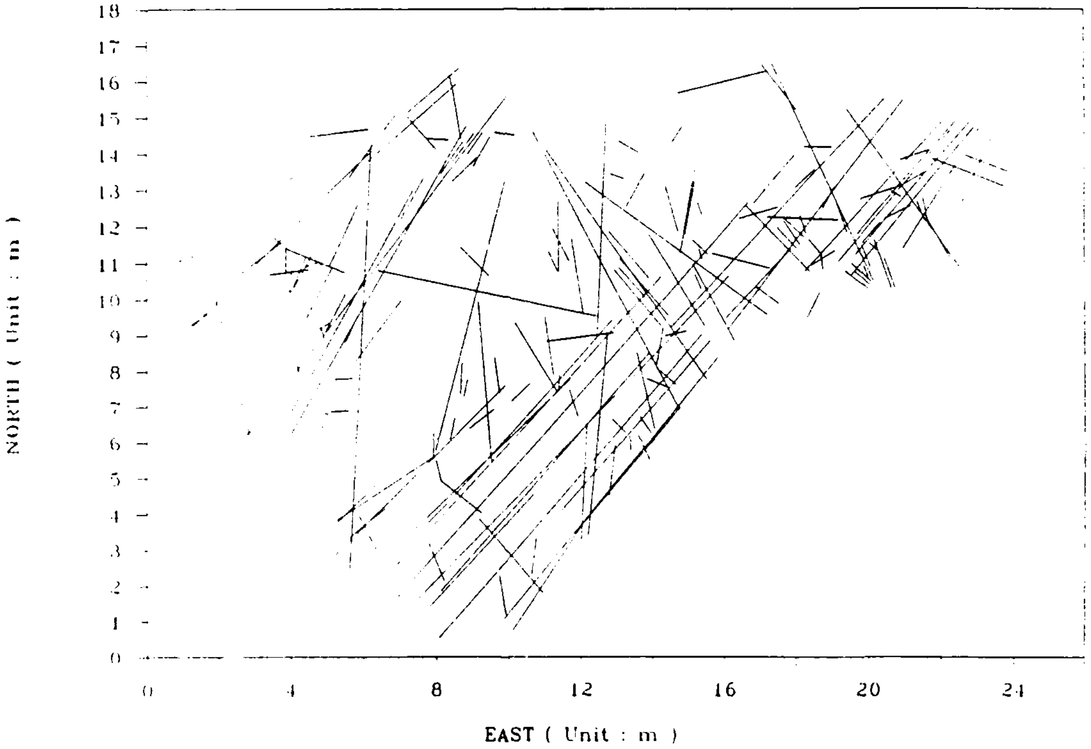


Figure 5-2: Digitized Trace Map of PA100

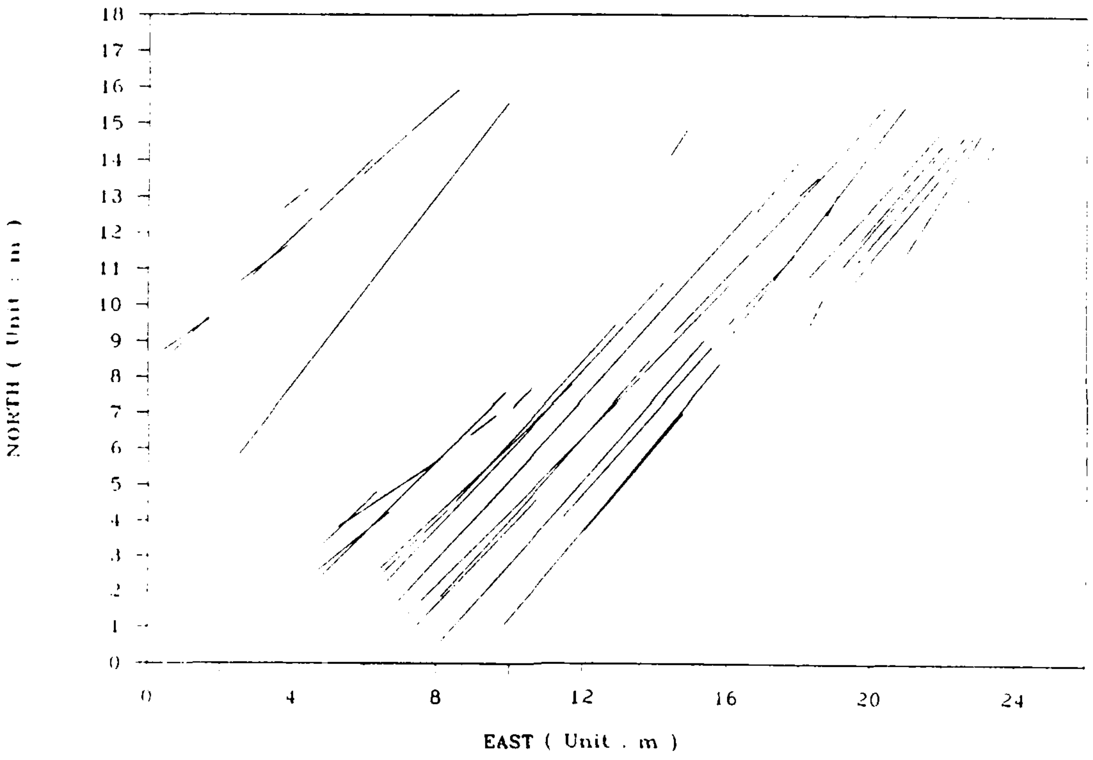


Figure 5-3: Digitized Trace Map of Set 1 (PA100)

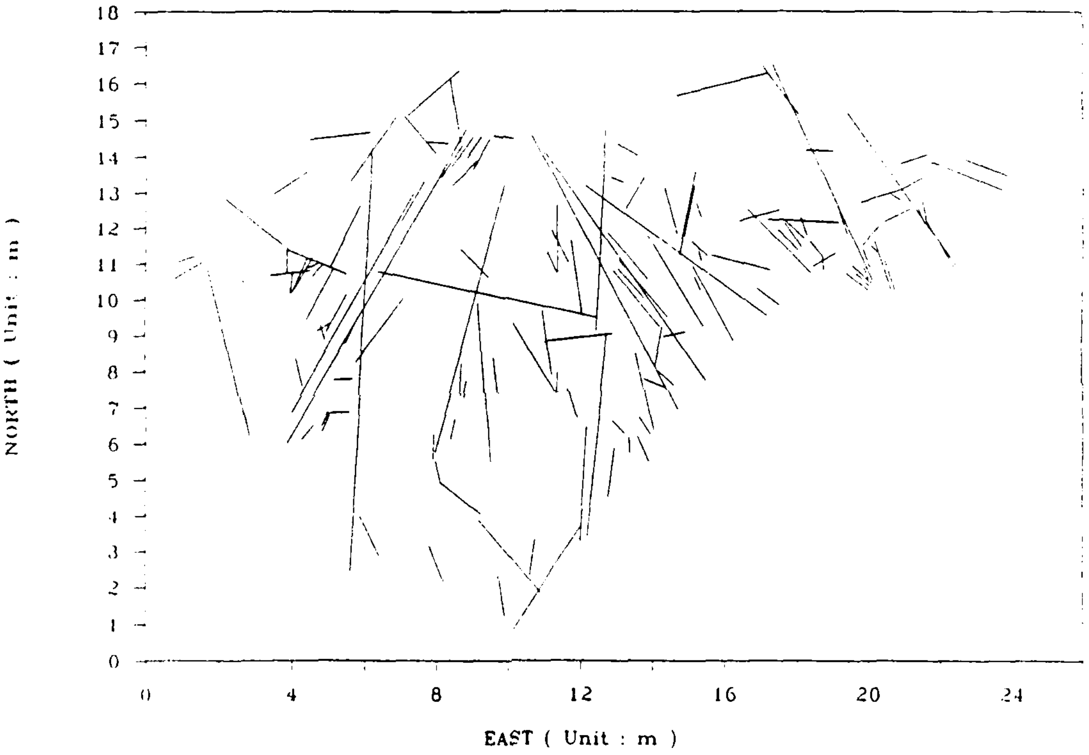


Figure 5-4: Digitized Trace Map of Set 2 (PA100)

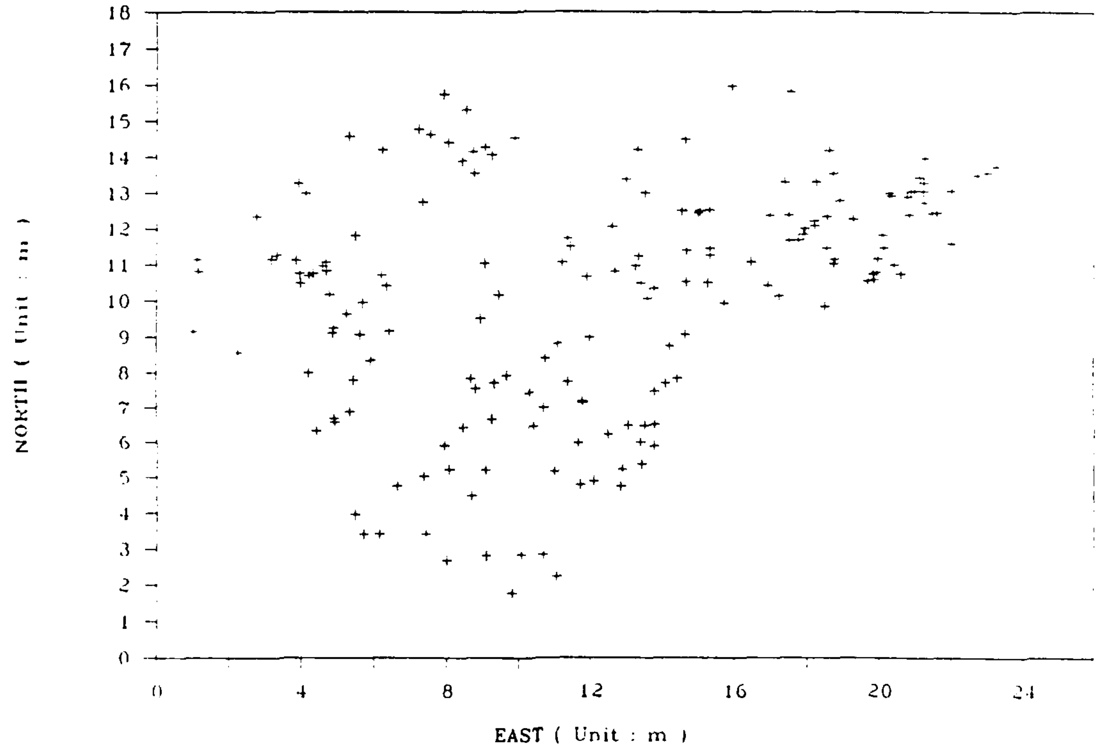


Figure 5-5: Mid Point Map of PA100 : Mid Points of All Traces

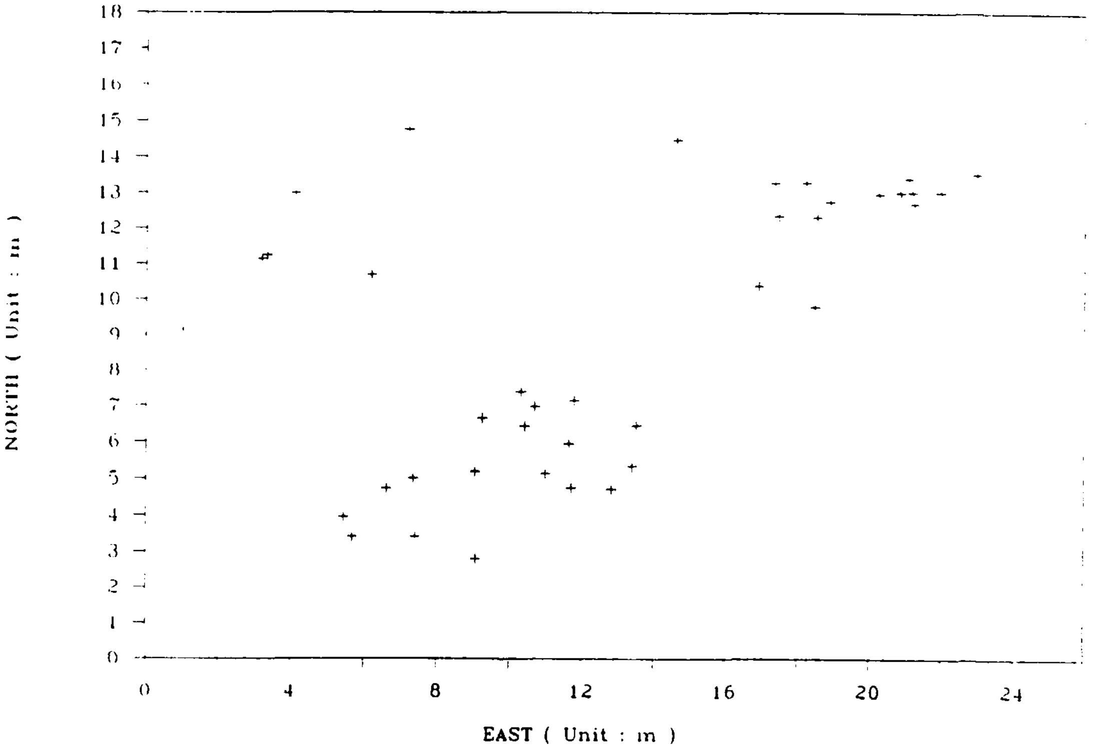


Figure 5-6: Mid Point Map of PA100 : Mid Points of Set 1 Traces Only

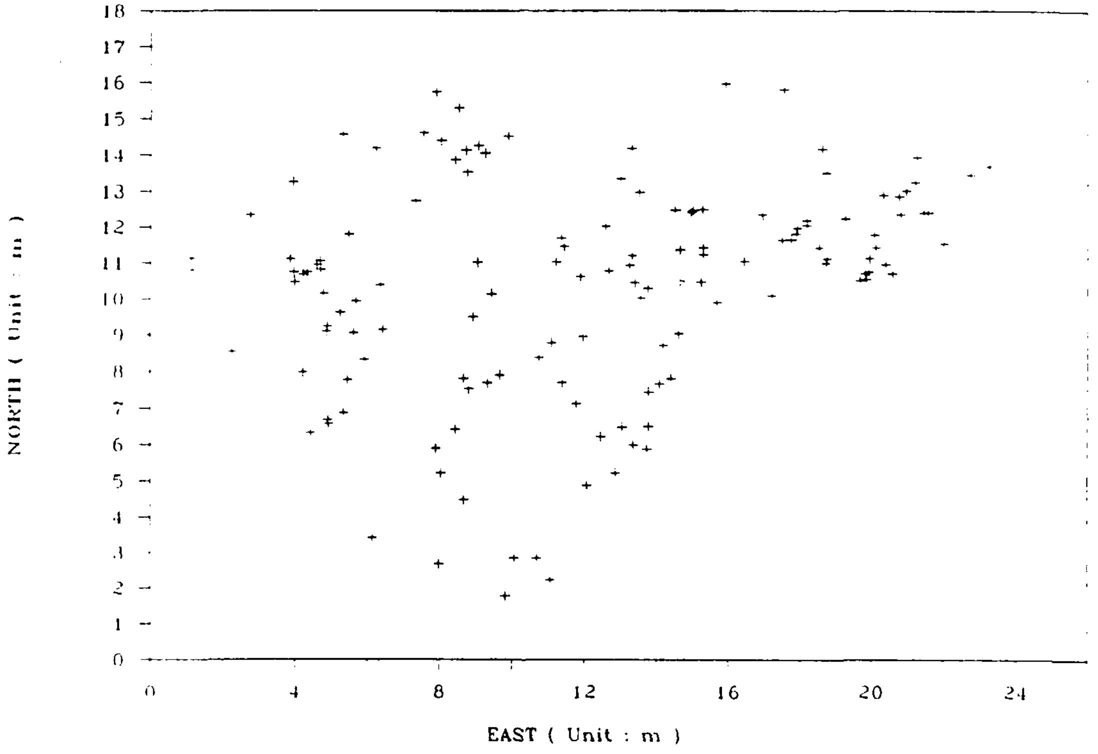


Figure 5-7: Mid Point Map of PA100 : Mid Points of Set 2 Traces Only

5.2 Step 1 - Midpoint Generation of Set 1

As can be seen in mid point map of set 1 (Fig. 5-6), both clustering and inhibitory behavior exist. Therefore, a possible alternative to a homogeneous, isotropic Poisson point pattern for set 1 might be a parent-daughter model. However, if we use the parent-daughter model, we might not be able to simultaneously represent the inhibitory and the clustered behavior, unless we use one more parameter such as inhibitory distance or we use a regionalized intensity function $\lambda(x)$ instead of a constant value λ as explained before in Chapter 4. We, therefore, use the doubly stochastic (Cox) point process to model fracture set 1 with the hierarchical model.

Fig. 5-8 shows a directional second moment of set 1 in which characteristic directions of 30-40° and 120-130° (counter-clockwise from the East) are used. In this particular case, if we confine our interests to small distances (e.g., up to 5m in Fig. 5-8), we cannot see any significant deviations from the isotropic case which is also shown in Fig. 5-8. The isotropic case is plotted for comparison and is identical to a homogeneous Poisson point process where the second moment, K(t), is identical to πt^2. This special situation of non-deviation from the isotropic case is quite probable since our mapped pattern has only 39 points, and we, therefore, cannot see any interactive behavior using the second moment. It also means that it is not possible to derive a correlation function directly because the correlation function becomes zero in an isotropic case. For this particular case, we modify the intensity function as follows.

$$\begin{aligned} \ln \lambda(x,y) &= \ln \lambda_0(y) + \varepsilon(x,y) \\ &\sim \sigma_0^2\, \rho_0(\Delta y) + \sigma_\varepsilon^2 \rho_\varepsilon(r) \end{aligned} \tag{5.1}$$

and

$$\begin{aligned} \ln \lambda_0(y) \sim C_0(\Delta x,\Delta y) &= \sigma_0^2 \rho_0(\Delta y) \\ \varepsilon(x,y) \sim C_\varepsilon(\Delta x,\Delta y) &= \sigma_\varepsilon^2 \rho_\varepsilon(r) \\ r = \sqrt{\Delta x^2+\Delta y^2} & \end{aligned} \tag{5.2}$$

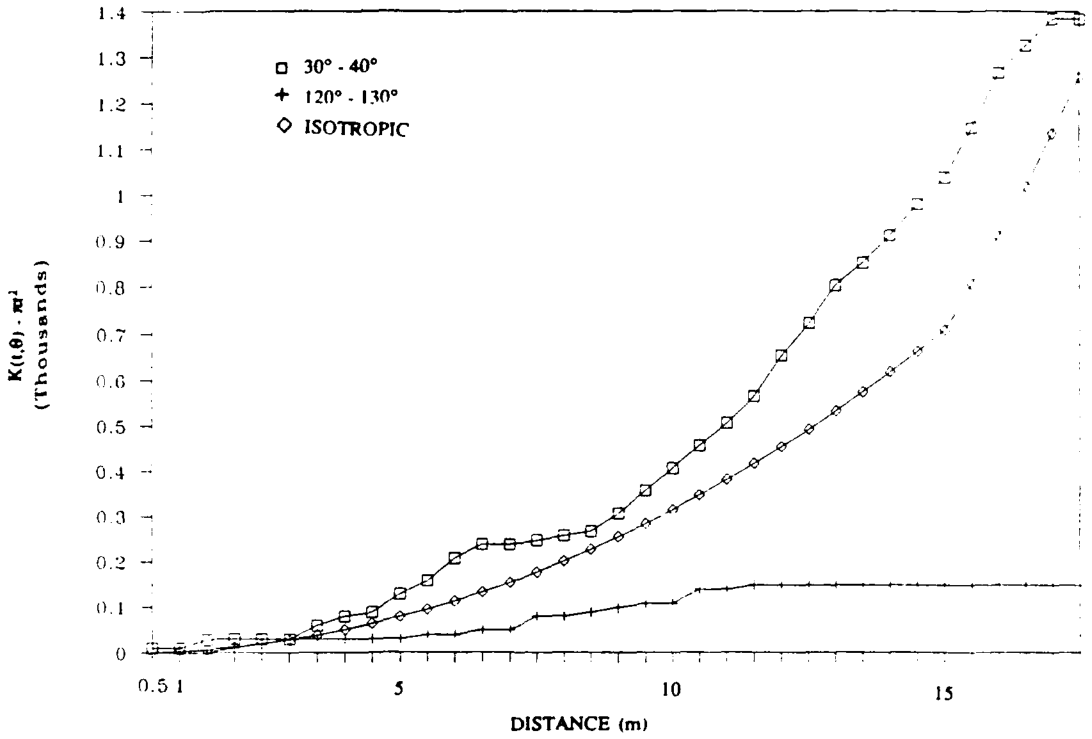


Figure 5-8: Directional Second Moment of Set 1. Characteristic directions of 30-40° and 120-130° are used to calculate the directional second moment. Isotropic case is also shown for comparison.

We assume that, due to the properties of a spectral density function, the mean intensity is governed only by the y direction and the clustering behavior is controlled by distance **r** (see Eqn (5.2)), where x and y are the transformed coordinates with the axes rotated by 40° (counter-clockwise from the East) and C(.) is an assumed covariance function.

With the Acceptance / Rejection scheme (Lewis and Shedler, 1979), we performed 50 simulations ; one of the simulations is shown in Fig. 5-9 where we assume that $\sigma_0^2 = \sigma_\varepsilon^2 = 0.5$, $\rho_\varepsilon(t) = 0.33t$ and $\rho_0(t) = 0.5t$. The second moment analysis of set 1 shows that a good similarity between the simulated and actual (mapped) point pattern exists (Fig. 5-10). The Monte-Carlo test also shows a satisfactory result. The Cox process can thus be used in a first step of the hierarchical model, particularly where both clustering and inhibitory behavior exist at the same time.

5.3 Step 2 - Traces of Set 1

The next step is to generate the traces for the simulated mid points of set 1. This simulation uses trace length statistics corrected for biases using the maximum likelihood formulations as derived in Chapter 4. In this case, set 1 has 15 *u* type traces (both ends observable), 22 *v* type traces (one end observable) and 2 *w* type traces (no ends observable). For simplicity, we neglect the *w* type traces.

The log likelihood of each trace type and the log likelihood of all traces are shown in Fig. 5-11, while a realization of the simulation is shown in Fig. 5-12.

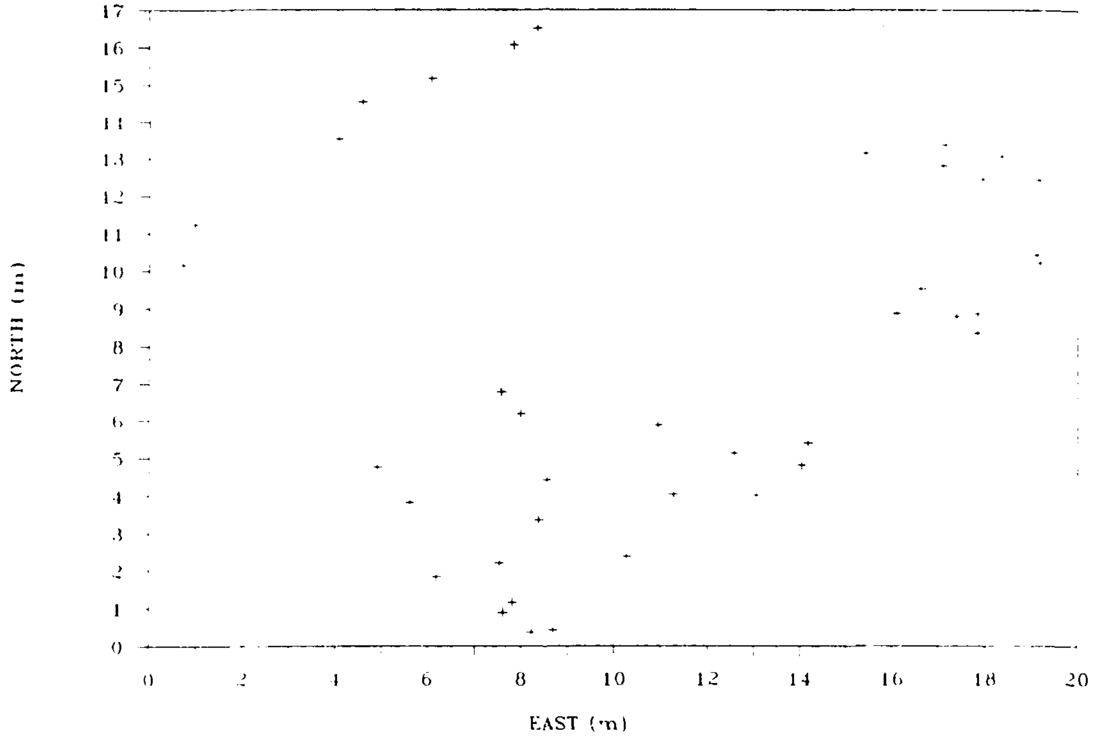


Figure 5-9: Simulation of Set 1 Mid Points with Cox Process

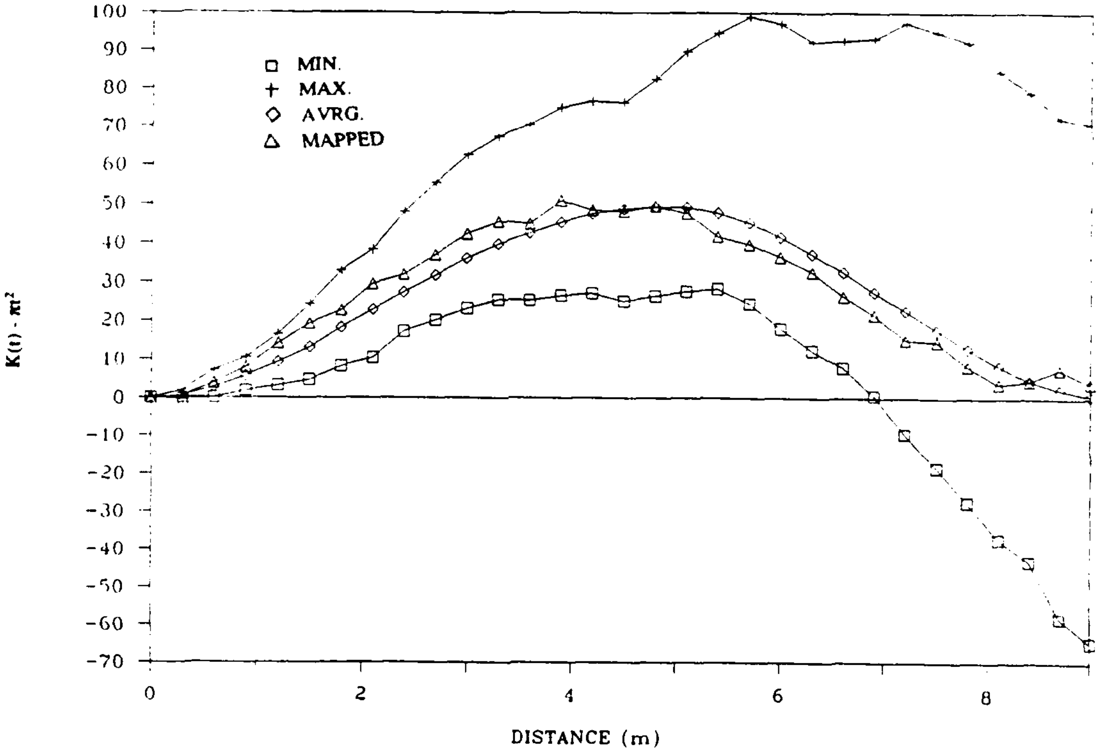


Figure 5-10: Second Moment Analysis of Set 1.
Acceptance band (MIN to MAX) and average value (AVRG) obtained with model simulations are shown. The second moment of the mapped data (MAPPED) is also given for comparison.

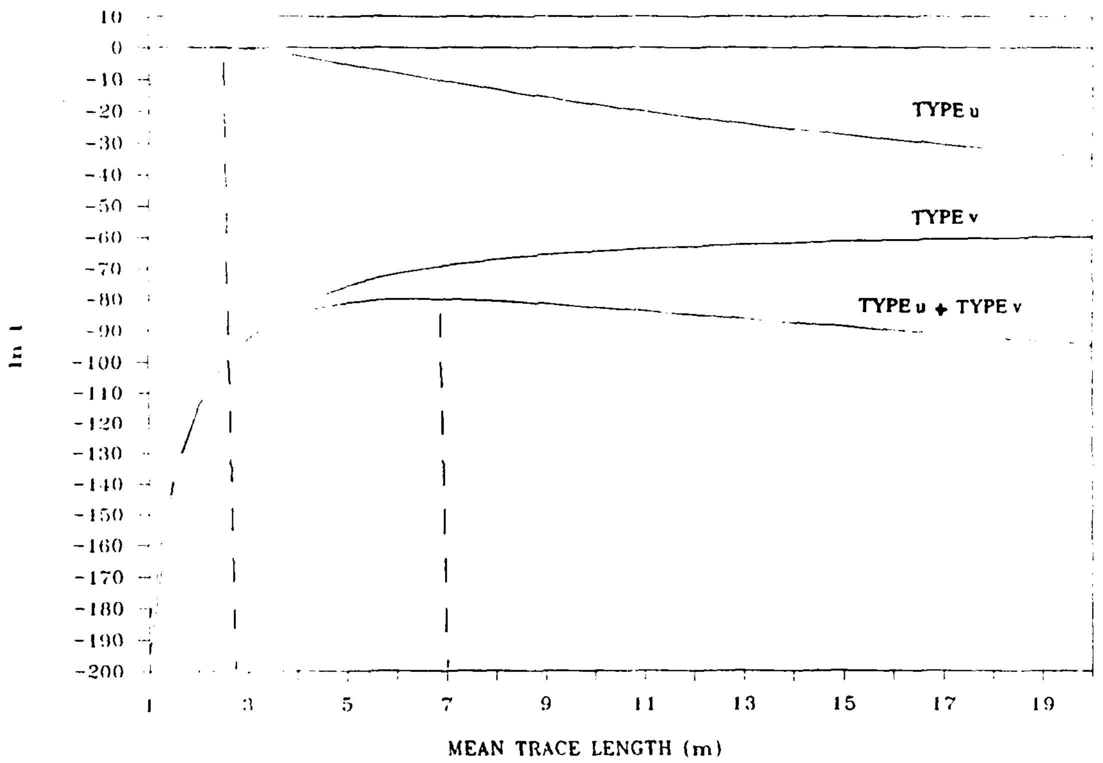


Figure 5-11: Log-Likelihood of Mean Trace Length for Set 1, assuming Exponential Distribution

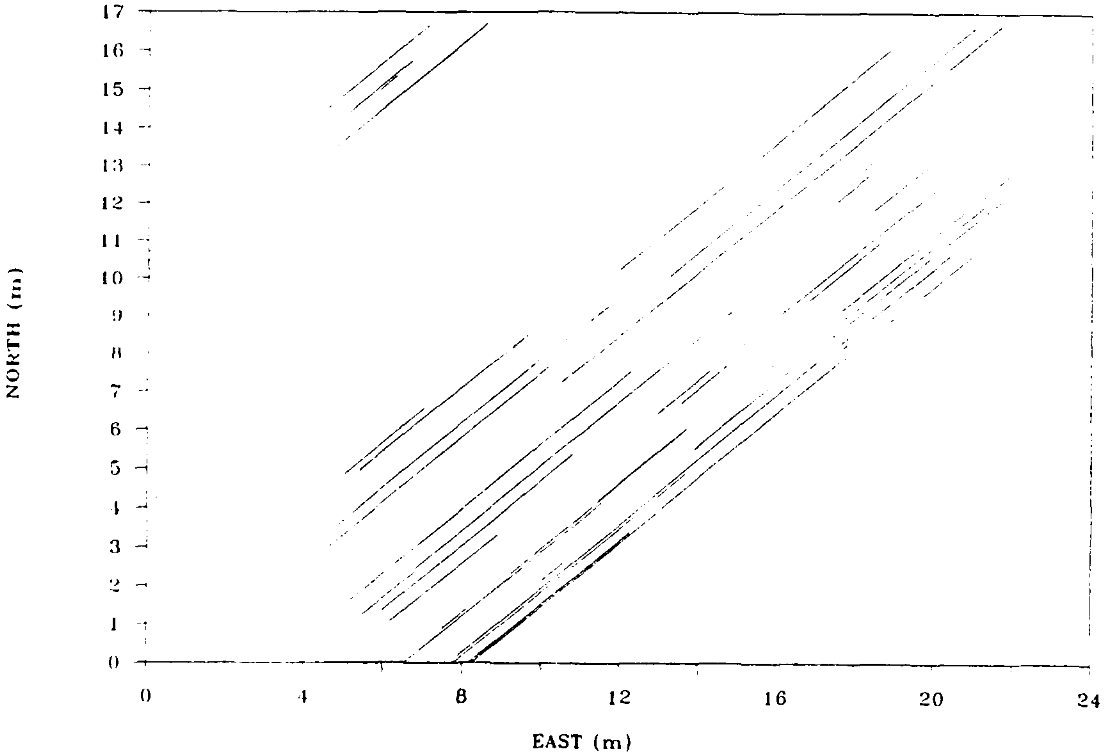


Figure 5-12: Simulation of Trace Lengths for Set 1

5.4 Step 3 - Independence Test

In step 3, dependence or independence between sets 1 and 2 has to be established. If there is no dependence between set 1 and set 2, we can generate each set by simply superimposing the traces set by set. One of two independence tests either the perturbation method or toroidal shift method can be used.

In the perturbation method, we calculate the (bivariate) second moment of the mapped pattern. We, next, perturb the mid points of set 2 by a small amount. This perturbation distance can be obtained using a random number generator. Then the second moment for the perturbed data is calculated. After a sufficient number of iterations, the significance (i.e., rank) of the second moment using the mapped pattern is estimated among those from perturbed data. Instead of perturbation, toroidal shifting of set 2 can be used as suggested by Lotwick and Silverman (1982). That is, instead of a small distance on a plane, a random distance on a toroid is used when perturbing the second set. Otherwise, The remaining procedure is the same as the perturbation method.

If the mapped pattern is composed of two independent Poisson point processes, where a Poisson process implies the stationary, isotropic behavior, the second moment measure will be as shown Fig. 5-13. For calculations in Fig. 5-13, we assume that the number of traces in set 1 is 39 and 138 in set 2. These numbers are the same as those in the mapped pattern of Fig. 5-5. As can be seen, the shifted bivariate second moment, $K(t) - \pi t^2$, is almost zero (i.e., independence) and satisfies the Monte-Carlo test since we assumed two independent Poisson point processes. However, with real mapped data, the shifted second moment is not zero and set 2 is positively correlated with set 1 (see Fig. 5-14). The Monte-Carlo test also rejects the independence assumption. From this result, we conclude that our mapped data set is neither homogeneous nor isotropic, and that it shows clustering. That is, set 2 has a tendency to get closer to set 1, or, in a rock mechanical sense, if there is a pre-existing fracture, the next one tends to be near the pre-existing fracture.

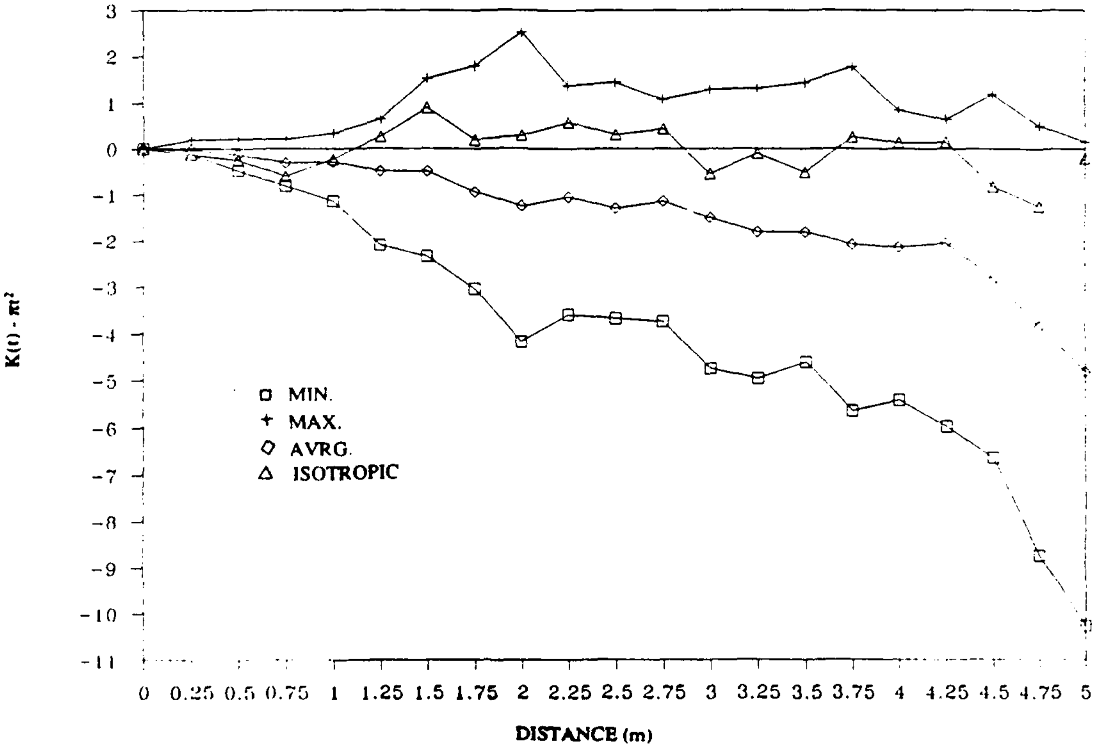


Figure 5-13: Independence Test with Simulated Homogeneous Poisson Point Process for Sets 1 and 2.
Acceptance band (MIN to MAX) and average value (AVRG) are constructed through perturbations. When the data is from an isotropic Poisson point process (ISOTROPIC), it will be inside the band.

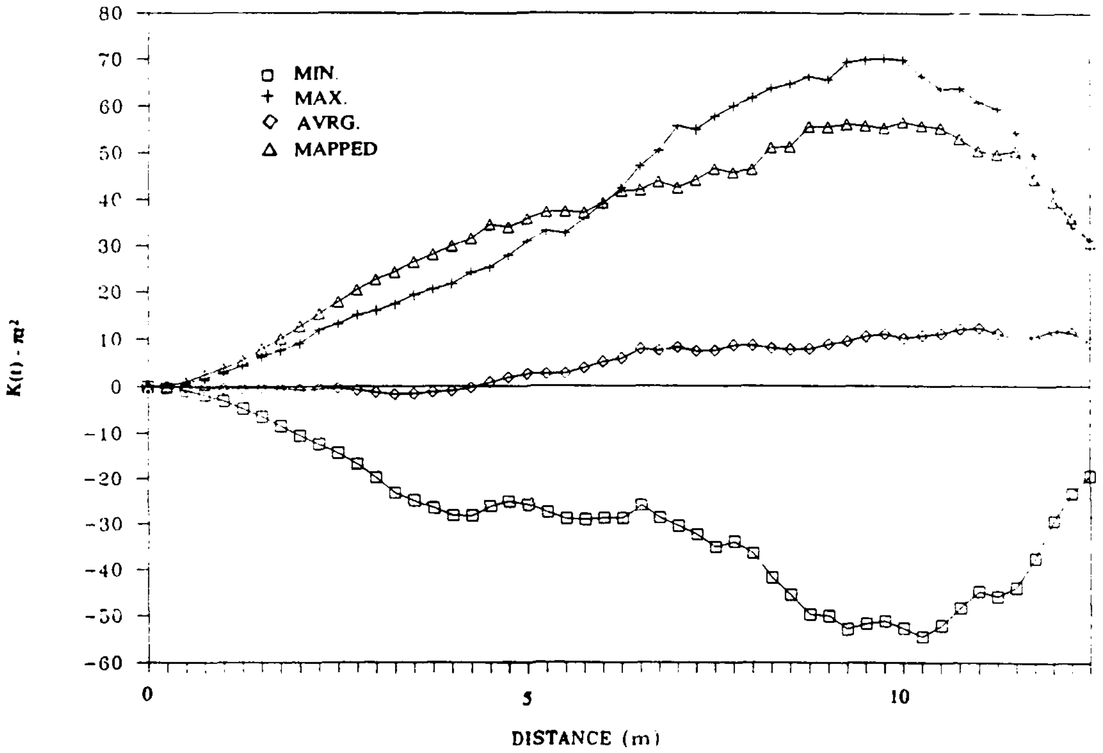


Figure 5-14: Independence Test of Mapped Pattern. Symbols are the same as those in Fig. 5-13. The independence of the actual mapped pattern (MAPPED) is tested. It shows the dependence of set 2 on set 1 traces.

5.5 Step 4 - Generation of Set 2 Midpoints

In the preceding steps, we have found that set 2 is not independent of set 1 and that there are positive correlations between the two sets. Thus, we use correlation measures when generating the mid point pattern of set 2. For this, we propose two possible correlation measure methods namely the line-kernel function method and the nearest neighbor fiber method.

When using the line-kernel function method, we divide each trace of set 1 into a number of segments of identical length, and we measure the center point of each segment. Next, we calculate the distance between a center point of a set 1 segment and a mid point of set 2. Using a bivariate normal distribution function, we derive a two-dimensional kernel estimate which has a circular distribution. This is done for all center points ; it produces the line-kernel estimates for our mapped data. To describe the relative uniformity of the pattern, we use a mixed distribution when calculating the MLE for conditional intensity :

$$f_k(x,y|\sigma,C)=\frac{C}{A}+\frac{1-C}{n_2}\sum_{1}^{n_1}\int K(x,y|\sigma)dL \qquad (5.3)$$

where, $f_k(x,y|\sigma,C)$ is a conditional density function mixing the uniform distribution and the line-kernel function, C and σ are parameters which can be estimated using MLE, A is an area, n_1 is the number of traces of set 1 and n_2 is the number of the mid points of set 2. The first part of the right-hand side of Eqn (5.3) comes from the uniform distribution of the pattern, and the second part is from the line-kernel function with which local clustering can be described.

Using MLE, we find C as 0.65 and σ as 0.75, and perform 50 simulations. One of the simulated mid point patterns is shown in Fig. 5-15. The goodness of fit is checked with two different methods. One is to use the bivariate second moment (K_{12}) with which the mid point patterns of set 1 and set 2 are compared. The bivariate second moment is shown in Fig. 5-16 ; good agreement between the mapped data and the simulated pattern is obtained. The other check is based on the (univariate) second moment (K_{22}) in which only the mapped data and simulated patterns of set 2 are compared (Fig. 5-17). This also shows acceptance of the simulated pattern. From these results, we conclude that the conditional intensity measures driven by the line-kernel function method are satisfactory. The mixture model using the line-kernel function is thus a suitable procedure to generate the mid point pattern of set 2 correlated with set 1.

Instead of the line-kernel function method, the nearest neighbor distance method can be used as a correlation measure. The main idea is to find the nearest fiber from a mid point of set 2. The nearest fiber distance can be measured between a mid point of set 2 and an arbitrary point in a fiber of set 1. Since the nearest distance can involve any point on a fiber of set 1, this scheme makes it possible to include the fiber length effect. To find the conditional intensity function, we again use a mixed distribution, i.e.,

$$f_n(x,y|C,\sigma) = \frac{C}{A} + \frac{1-C}{n_2} G(x,y|\sigma) \tag{5.4}$$

where $f_n(x,y|C,\sigma)$ is a conditional intensity function with parameters C and σ. $G(x,y|\sigma)$ is a nearest neighbor distance distribution function which is assumed to be normally distributed. Again, we assume that the conditional intensity function is mixed with the uniform distribution. From MLE, we found C = 0.65 and σ = 0.2. One of the simulated mid point patterns of set 2 is shown in Fig. 5-18. As we did with the line-kernel function method, we compared the simulated patterns with mapped data. Fig. 5-19 is a bivariate second moment analysis of sets 1 and 2, and Fig. 5-20 is a (univariate) second moment analysis of set 2. These results also show an acceptable goodness of fit of the mixture model driven by the nearest neighbor fiber method.

The advantage of the nearest neighbor function method is that it only considers a point within a fiber which becomes a nearest neighbor, while the line-kernel function

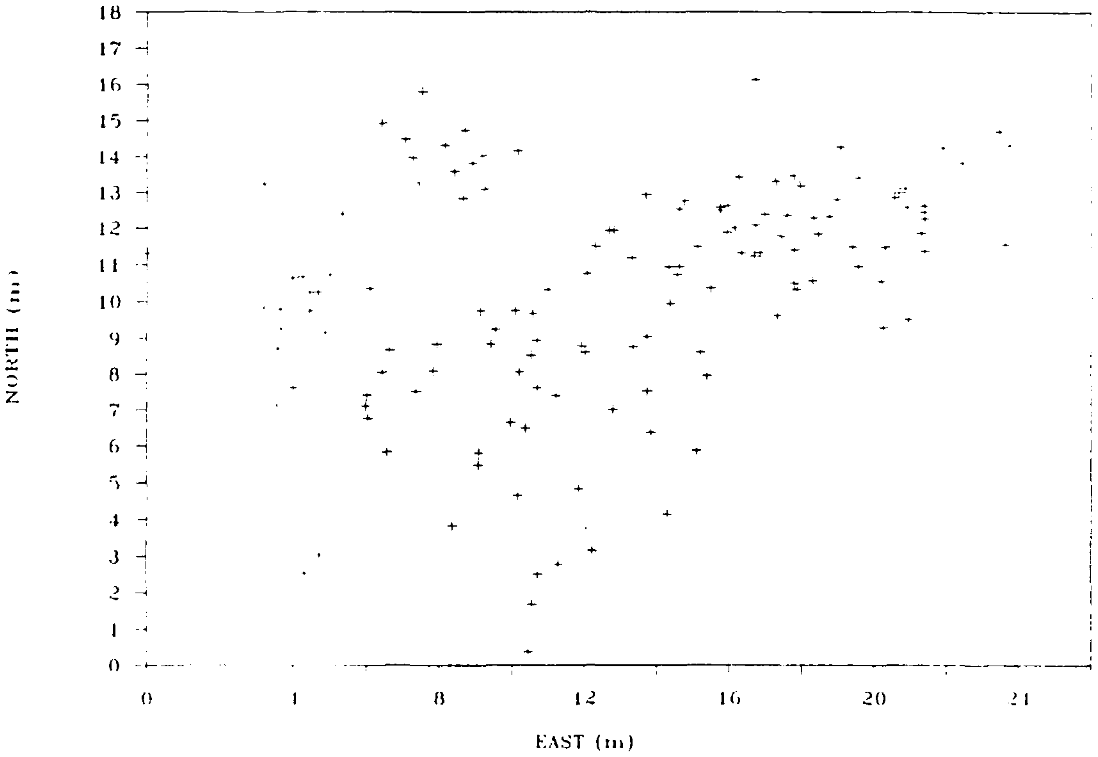


Figure 5-15: One of the Simulated Mid Point Patterns of Set 2 Using the Line-Kernel Function Method to Express Correlation.

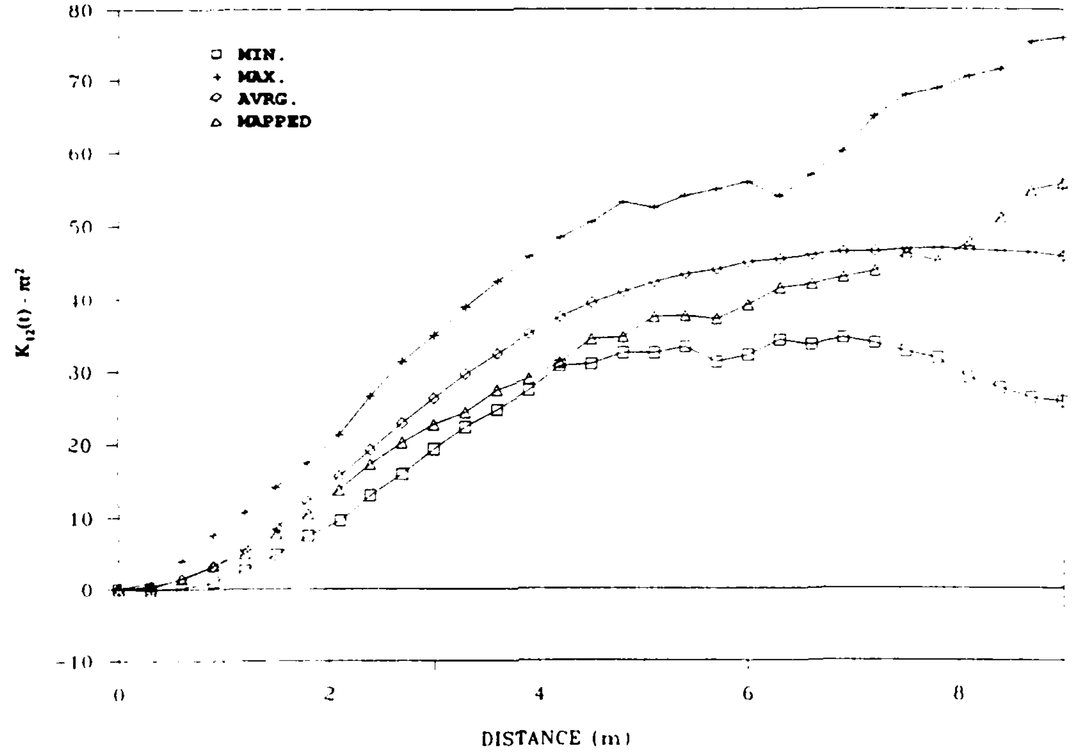


Figure 5-16: Bivariate Second Moment Analysis for Sets 1 and 2 - Mid Point Correlation Based on the Line-Kernel Function Method. Acceptance band (MIN to MAX) and average value (AVRG) are obtained with model simulations, while the mapped pattern (MAPPED) is from actual data.

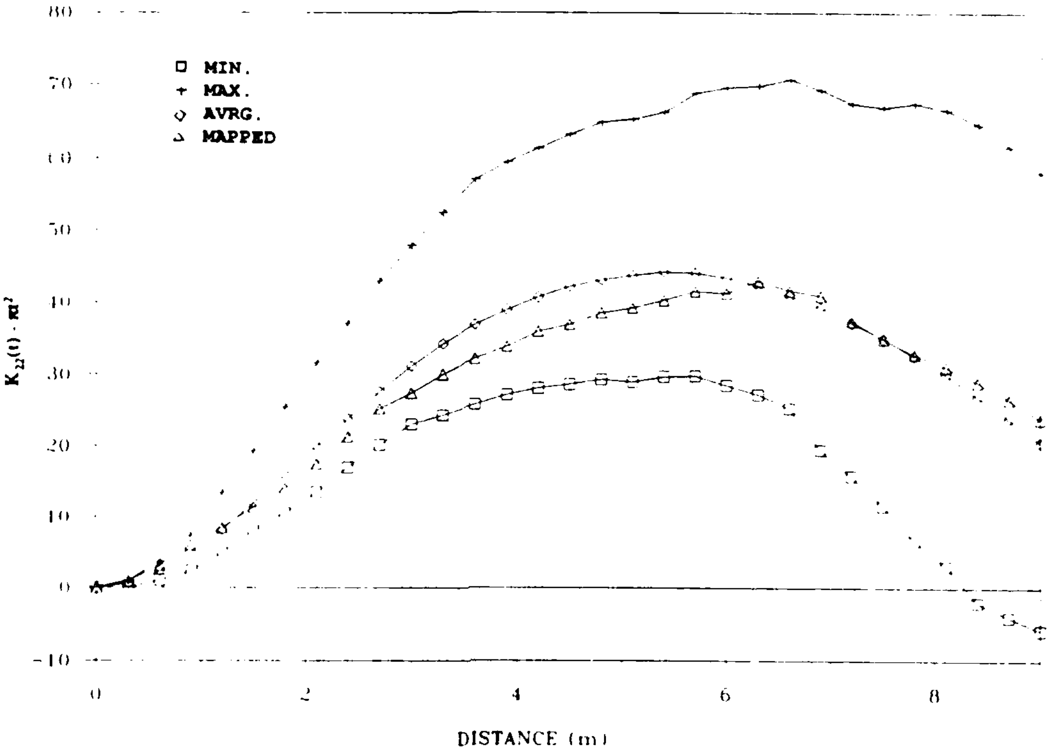


Figure 5-17: (Univariate) Second Moment Analysis for Set 2 Mid Points Using the Line-Kernel Function Method.
Symbols are the same as those in Fig. 5-16.

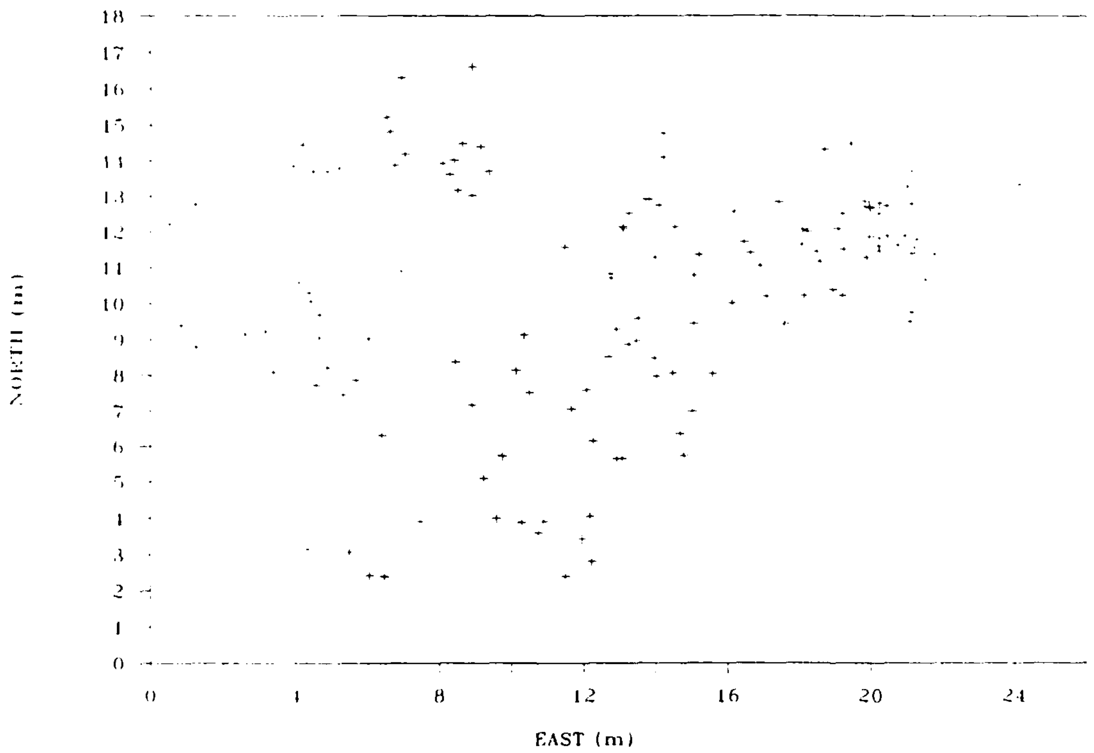


Figure 5-18: One of the Simulated Mid Point Patterns of Set 2 Using the Nearest Neighbor Function Method to Express Correlation

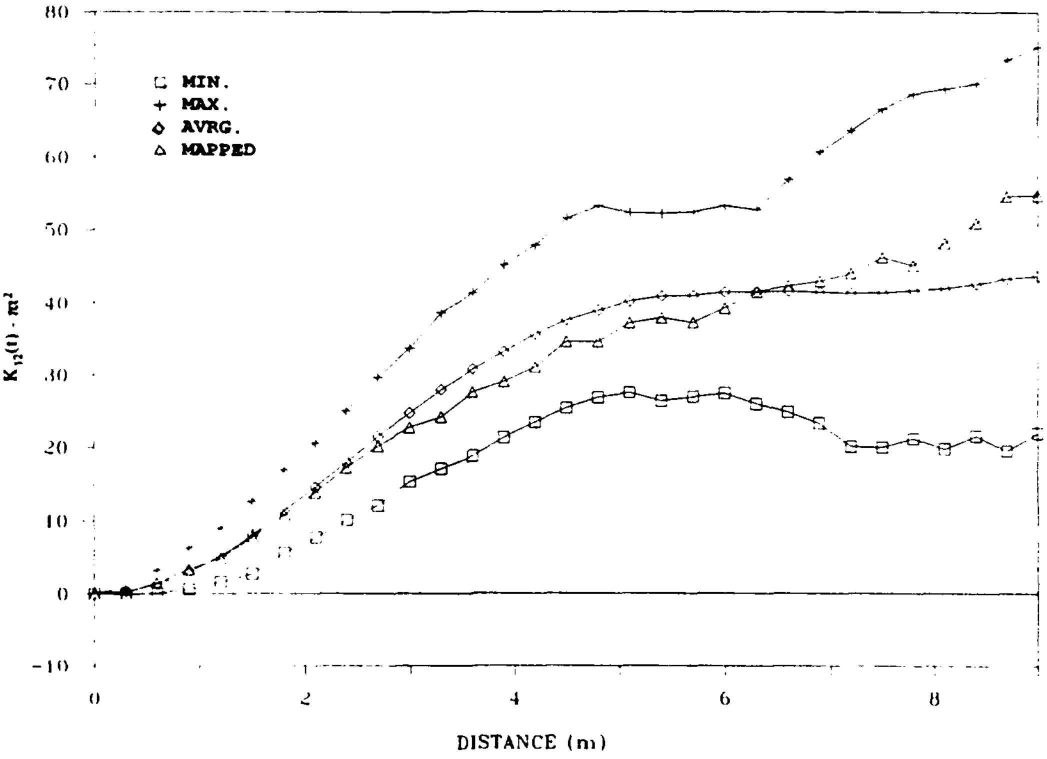


Figure 5-19: Bivariate Second Moment Analysis for Sets 1 and 2 - Mid Point Correlation Based on the Nearest Neighbor Function Method. Acceptance band(MIN to MAX) and average value (AVRG) are obtained with model simulations, while the mapped pattern (MAPPED) is from actual data.

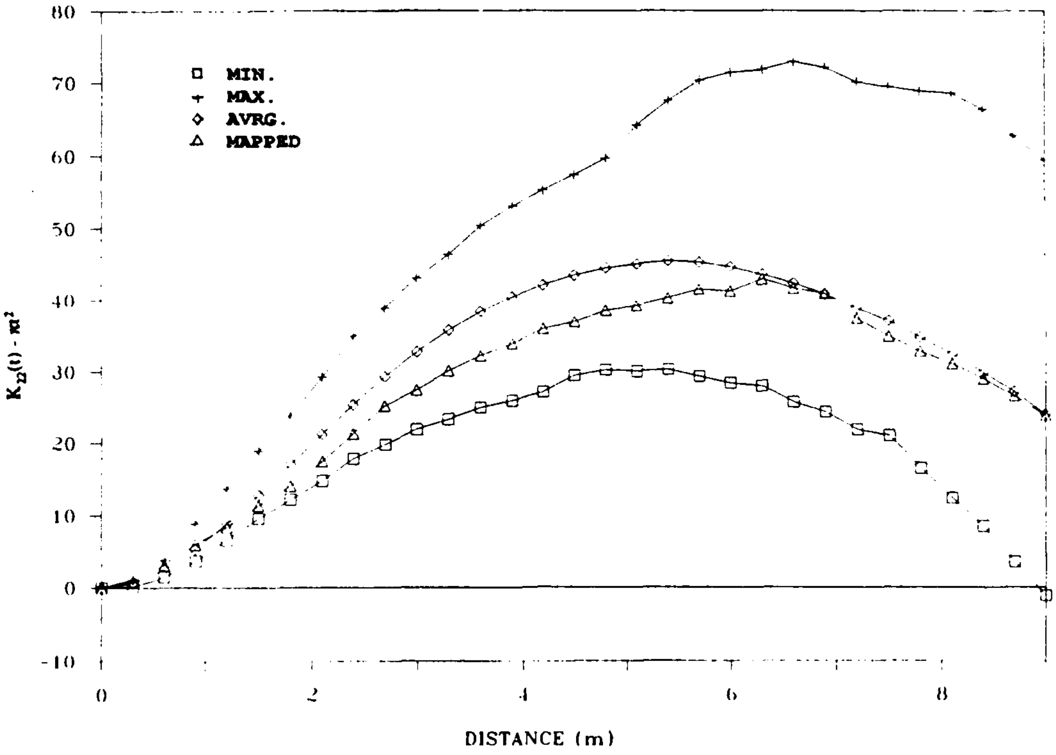


Figure 5-20: (Univariate) Second Moment Analysis for Set 2 Mid Points Using the Nearest Neighbor Function Method.
Symbols are the same as those in Fig. 5-19.

method requires an additional step when calculating the line-kernel function. That is, we need to calculate the kernel estimates of all the segments in a fiber.

5.6 Step 5 - Generation of Set 2 Traces

Once we have a mid point pattern of set 2, we determine the trace length and orientation for each point of set 2 :

First, the mean trace length can be calculated using MLE as described before for set 1. With MLE, we estimate the mean trace length to be 2.5m assuming that there are no trace length correlations between sets 1 and 2.

To find the orientation correlations, we again consider the nearest fiber concept. Since we have assumed that the orientation of set 1 is constant (fixed at 40°, see Fig. 5-3), we calculate orientation distributions of set 2 conditional on set 1. Fig. 5-21 shows the orientation distribution of set 2 without considering the correlation measures. When we calculate the orientation distributions of set 2 as a function of the nearest neighbor distance to set 1, we find that there are no correlations between the sets. This is evident from Fig. 5-22 where the orientation distributions of set 2 obtained by the nearest neighbor distance to set 1 (1m, 2m and 3m) are plotted. Also, the distribution of Fig. 5-21 is plotted for comparison. Thus, we only need an orientation distribution function for set 2 without any conditions. For this, we applied two distribution functions which have distribution forms on a circle. One is the Von Mises distribution (Eqn (5.5)) and the other is the wrapped normal distribution (Eqn (5.6)) (see Mardia, 1972 or Fisher, et al, 1987) :

$$f(\theta)=\frac{1}{2\pi I_o(\kappa)}\exp(\kappa\cos(\theta-\alpha)), \qquad 0\leq \theta < 2\pi \tag{5.5}$$

With

$$I_o(\kappa)=1+\frac{(0.5\kappa)^2}{1!1!}+\frac{(0.5\kappa)^4}{2!2!}+\frac{(0.5\kappa)^6}{3!3!}+\cdots$$

where, κ is a shape parameter and α is a location parameter.

$$f(\theta)=[\sigma\sqrt{2\pi}]^{-1}\sum_{-\infty}^{\infty}\exp(-\frac{(\theta-\alpha-2r\pi)^2}{2\sigma^2}), \quad 0\leq\theta<2\pi \tag{5.6}$$

where, α is a location parameter and σ^2 is a shape parameter.

To fit the data, we use the bi-modal form of the orientation distribution function, i.e., for the Von Mises distribution,

$$\begin{aligned} f(\theta) &= \frac{C}{2\pi I_o(\kappa)}\exp(\kappa\cos(\theta-\alpha)) \\ &+\frac{1-C}{2\pi I_o(\kappa)}\exp(\kappa\cos(\theta-\alpha-\pi)) \end{aligned} \tag{5.7}$$

and for the wrapped normal distribution,

$$\begin{aligned} f(\theta)&=C[\sigma\sqrt{2\pi}]^{-1}\sum_{-\infty}^{\infty}\exp(-\frac{(\theta-\alpha-2r\pi)^2}{2\sigma^2}) \\ &+(1-C)[\sigma\sqrt{2\pi}]^{-1}\sum_{-\infty}^{\infty}\exp(-\frac{(\theta-\alpha-\pi-2r\pi)^2}{2\sigma^2}) \end{aligned} \tag{5.8}$$

In our case, C is 0.5. Using MLE, we find the parameters for the Von Mises distribution to be α = 105° and κ = 1.9, and for the wrapped normal distribution α = 105°, σ = 0.8. Fig. 5-23 shows the orientation distribution forms of these two functions as well as the orientation distribution of set 2. To evaluate the appropriateness of these models relative to the mapped data, we performed a Kolmogorov-Smirnov test for each case. Both distributions fit the data at the 5% significance level. Both models can thus be used to simulate the pattern of set 2. Fig. 5-24 is a simulated pattern of set 2 with the mid point pattern obtained by conditional intensity functions (see, Section 5.4) and with the Von Mises orientation distribution. Fig. 5-25 is the same except that the orientation distribution is the wrapped normal distribution.

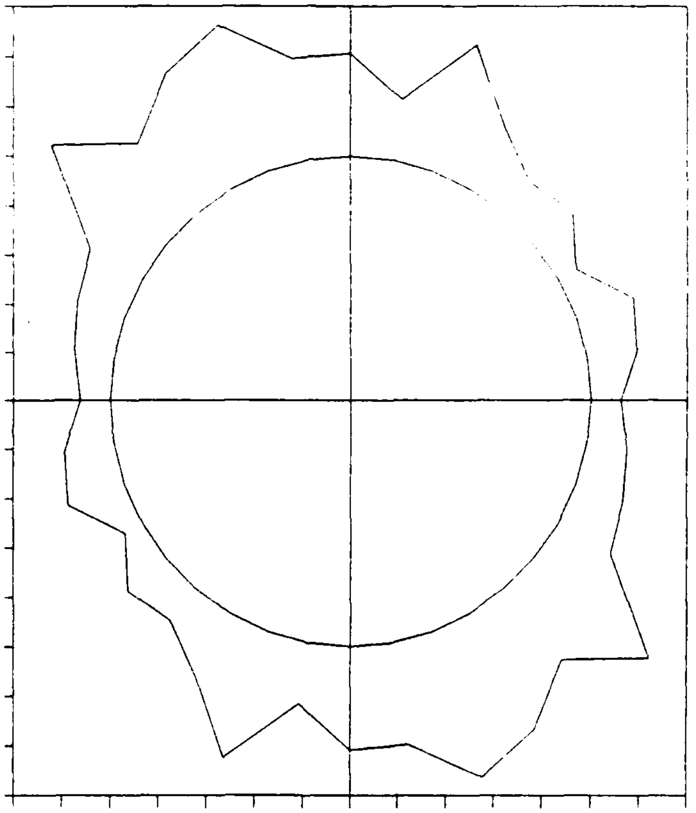


Figure 5-21: Orientation Distribution of Set 2 on a Circle

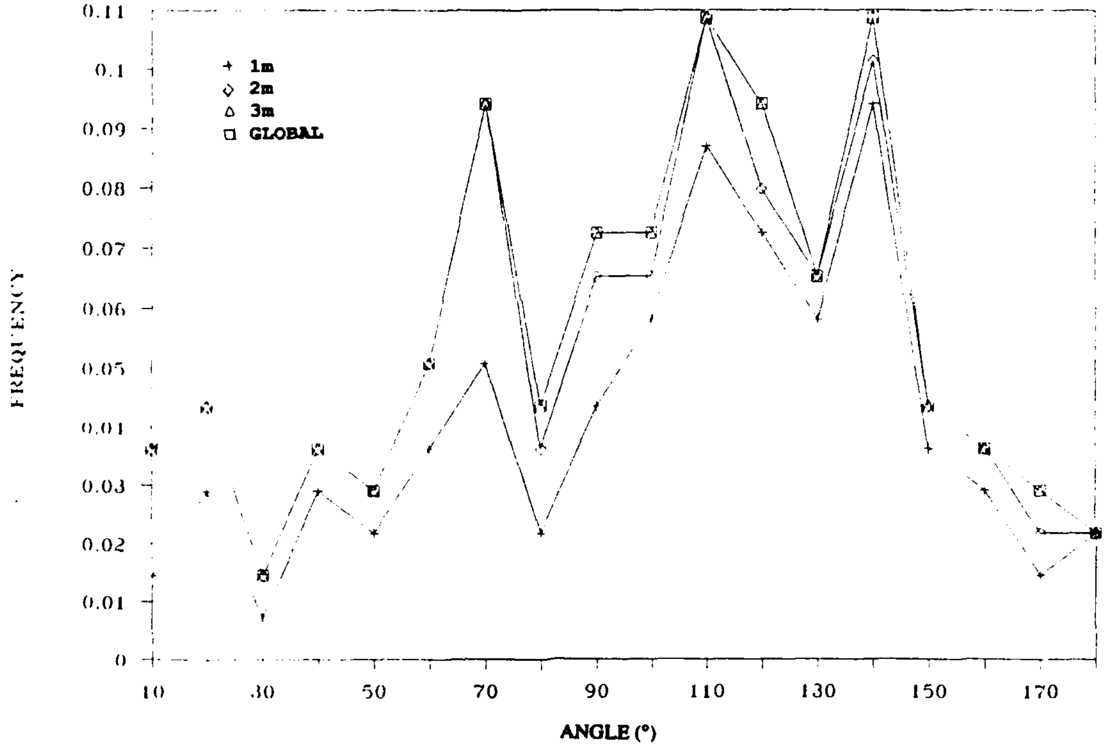


Figure 5-22: Orientation Distribution of Set 2 Based on a Nearest Fiber Distance of Set 1. Orientations are measured with increasing distances (1m, 2m, 3m), also Fig. 5-21 (GLOBAL = actual orientaion data) is shown for comparison.

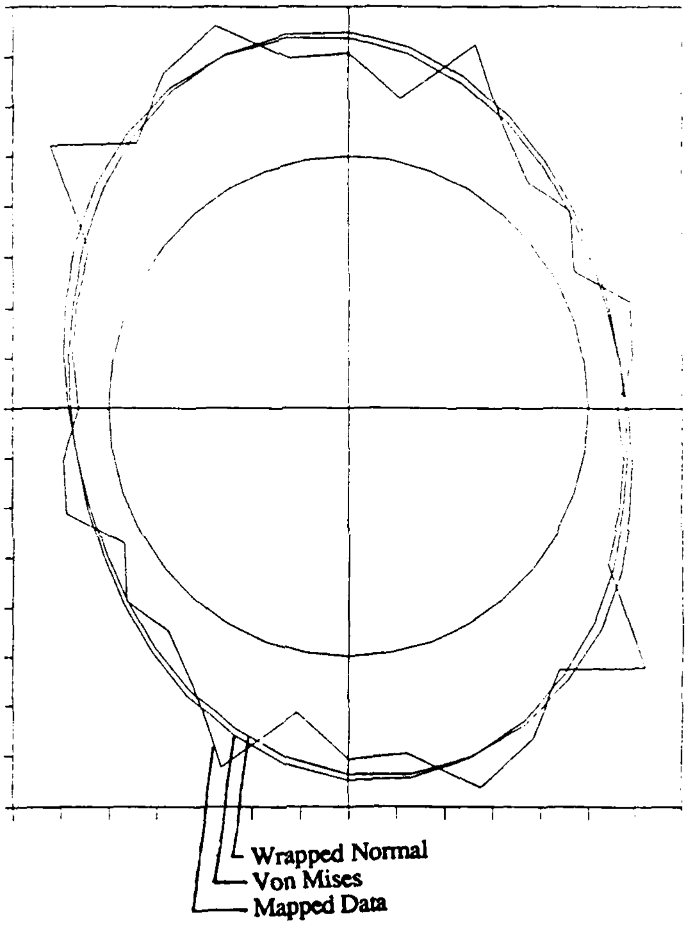


Figure 5-23: Wrapped Normal and Von Mises Orientation Distribution Fitted to Orientation Data of Set 2

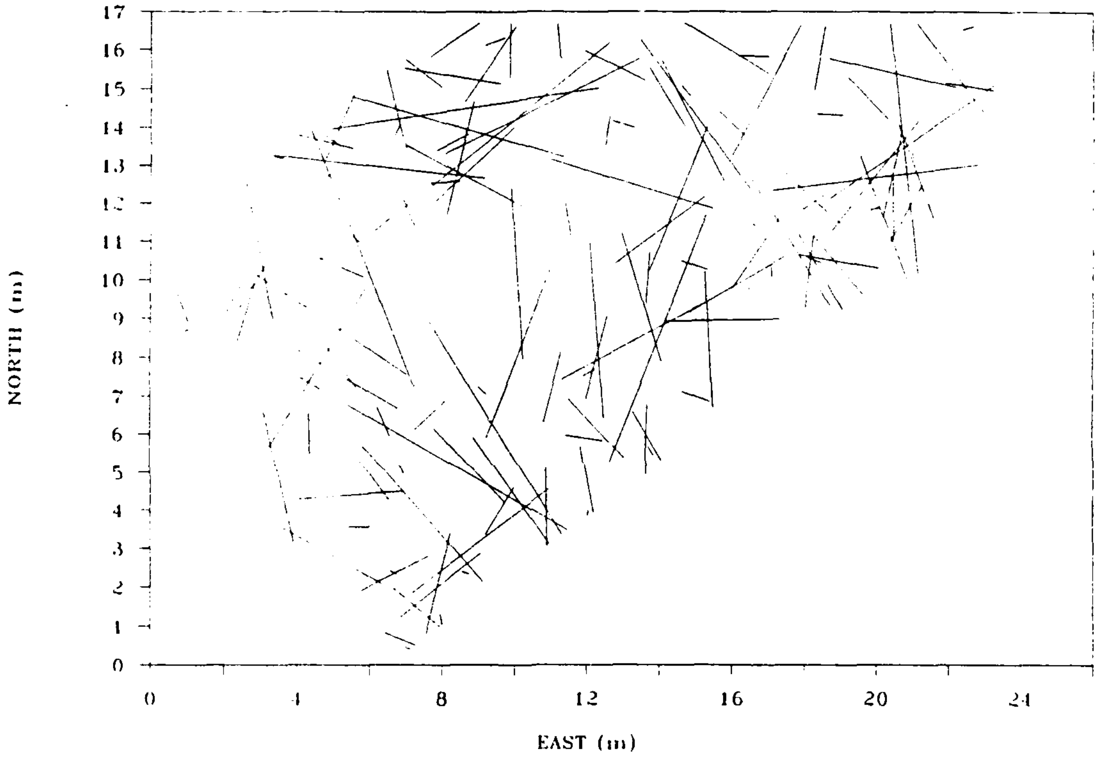


Figure 5-24: Trace Pattern of Set 2 with Von Mises Orientation Distribution

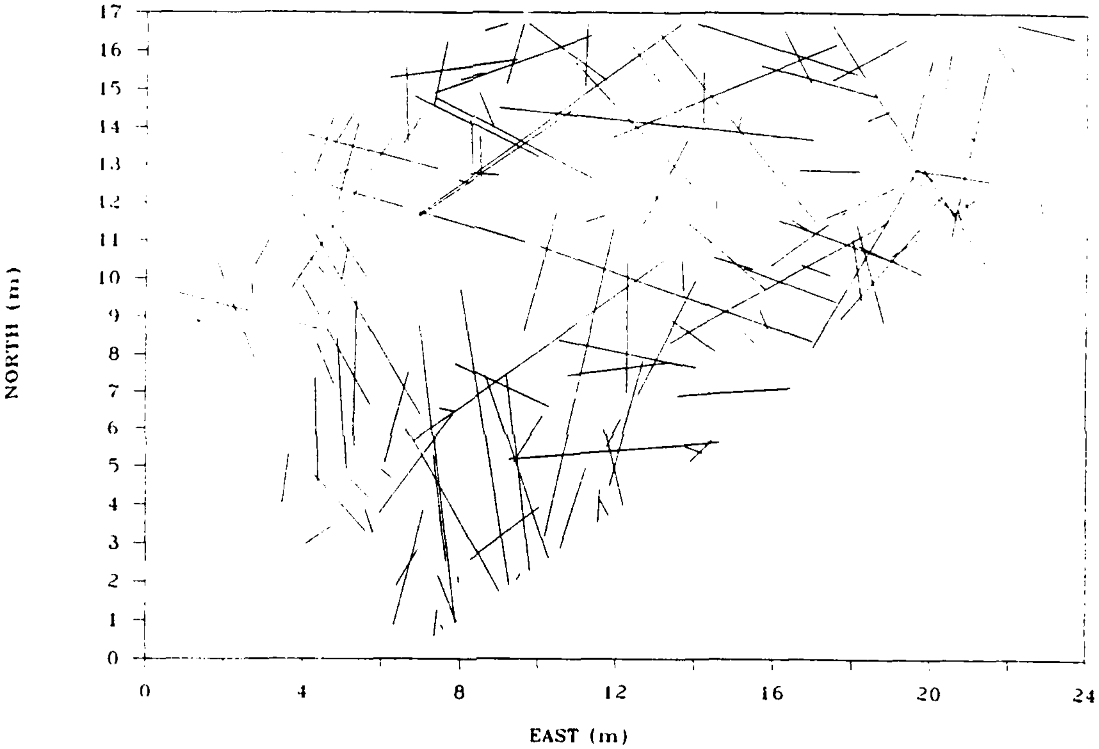


Figure 5-25: Trace Pattern of Set 2 with Wrapped Normal Distribution

5.7 Step 6 - Termination

We now have the mid point pattern, mean trace length and orientation distribution of sets 1 and 2. The remaining issue is to define a termination probability among traces. Dershowitz and Einstein (1988) defined the termination probability by counting both the intersection points and the termination points among traces. An intersection point can be a crossing point or a termination point as illustrated in Fig. 5-26 (a) and (b). The Dershowitz/Einstein termination probability is simply the ratio of termination points / intersection points. However, in our mapped pattern, this method is not appropriate because traces of set 2 cross traces of set 1 at many intersection points, i.e., an intersection of a set 2 trace with a set 1 trace is not necessarily a termination. To express the termination probability more appropriately, a new definition is introduced here. All termination points and crossing points are counted except the points between ihe center of a set 2 trace and the farthest intersection point. Thus, in Fig. 5-26 (d), each trace of set 2 ($t_{[2]}$ has a maximum of two intersection (crossing or termination) points (A and B) ; it can also have only one or none. Using only these corrected intersection points, we derive termination probability as the ratio of termination points / intersection points. This procedure introduces some distortion in the distribution of set 2 trace lengths ; however, the resulting bias does not appear to be serious. Fig. 5-26 illustrates the determination of this termination probability. If the pattern is not complicated, the termination probability with our new termination scheme is the same as that of Dershowitz and Einstein (1988) as in Fig. 5-26 (c), but if the pattern is complex, the new termination probability will be different (Fig. 5-26 (d)). This termination probability is used when simulating the pattern.

We have now reached the point where the modeled fracture pattern can be completed. The simulated patterns of PA100 with the Von Mises orientation distribution function is shown in Fig. 5-27 and with the wrapped normal distribution function in Fig. 5-28.

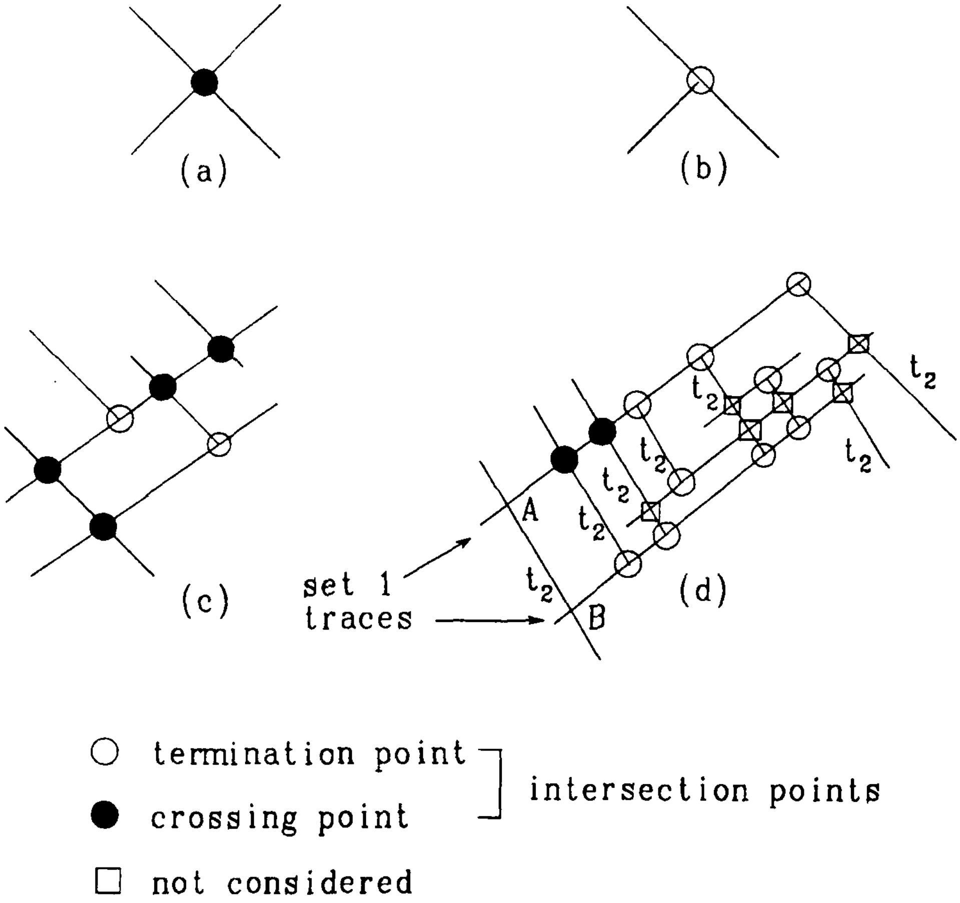


Figure 5-26: Calculation of Termination Probability

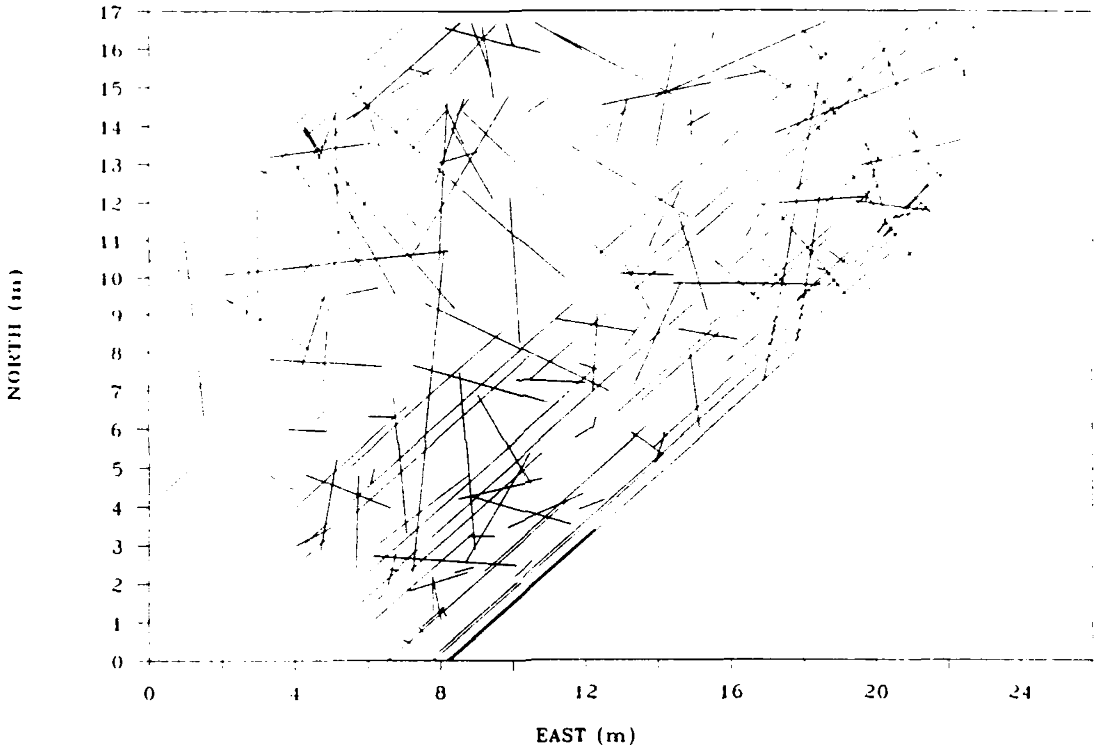


Figure 5-27: One of the Simulated Patterns of PA100 with the Von Mises Orientation Distribution Function

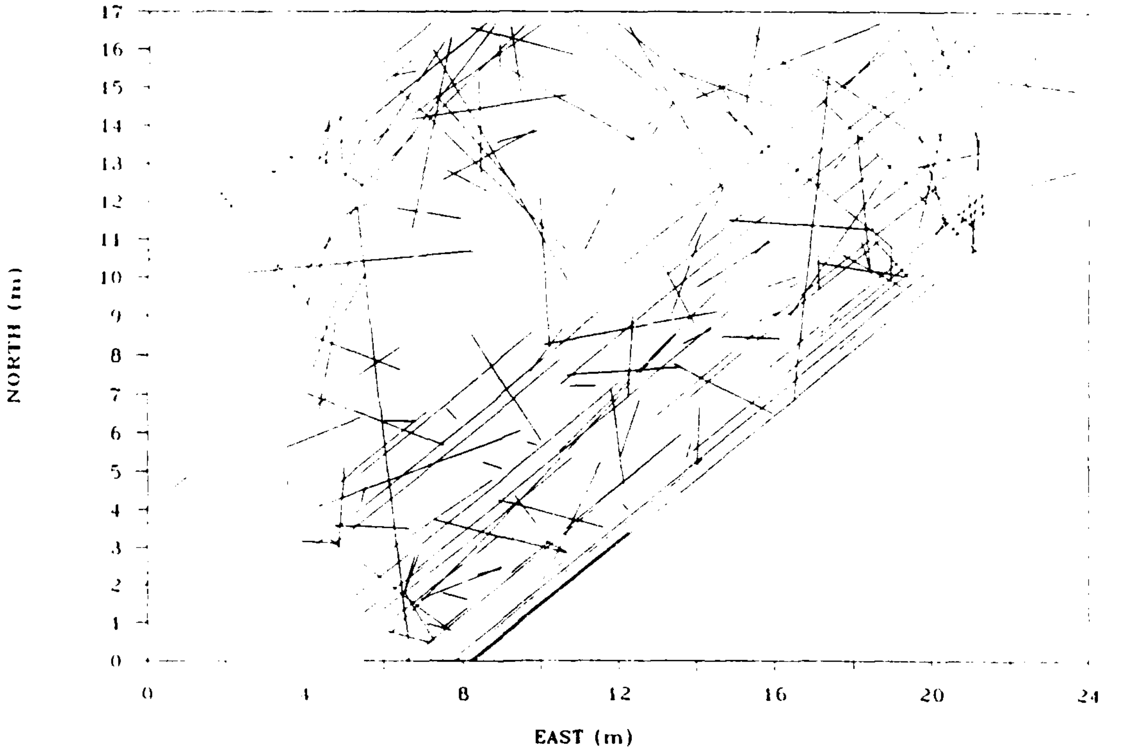


Figure 5-28: One of the Simulated Patterns of PA100 with the Wrapped Normal Orientation Distribution function

5.8 Concluding Comments

In applying the fiber process to model fracture trace distributions, we did the following :

1. We divide all the traces of a map into two sets according to orientation and we define sets 1 and 2.
2. A doubly stochastic point (Cox) process is used as a model for the mid point pattern of set 1.
3. With a new MLE scheme, we calculate the mean trace length for each set assuming an exponential trace length distribution.
4. Using the bivariate point process, we find that mid points of set 1 and set 2 are dependent on each other and that they are positively correlated.
5. To calculate the conditional intensity of set 2, two correlation measure methods are used. One is the line-kernel function method and the other is the nearest neighbor fiber method. Using these two schemes, we simulate the mid point pattern of set 2 (the nearest neighbor fiber method is easier to use than the line-kernel function method).
6. The Von Mises and the wrapped normal orientation distributions on a circle are proposed as models for set 2, whereas a fixed orientation of 40° is used in set 1.
7. To consider the terminations among traces a new termination probability is proposed such that a trace is terminated at the farthest end with a prescribed probability.

In Chapter 4 and 5, we proposed our hierarchical fiber model and studied its feasibility. It showed that fracture traces of several dependent or independent sets can be modelled with the hierarchical procedure. The models can represent clusters as well as all the other geometrical characteristics. The application discussed in this chapter related to only two sets but the hierarchical model can be just as easily applied to a larger number of sets.

Chapter 6

TOPOLOGICAL APPLICATION OF FRACTURE GEOMETRY MODEL : SLOPE STABILITY ANALYSIS

6.1 Introduction

When modelling the performance of fractured rock, be that slope stability, deformation under foundations and around tunnels or flow through the rock mass, both the geometry and the mechanics need to be represented. As was shown in preceding chapters, it is now possible to represent the two dimensional stochastic geometry by the hierarchical fracture model. Also, developments toward a three dimensional hierarchical fracture modelling are under way.

What will be discussed in this Chapter is the combination of the fracture geometry model with suitable mechanical models to represent slope stability. As will be discussed in more detail below, the mechanical models for slope stability involve models representing sliding along discontinuities (fractures) and models representing the creation of new fractures in intact rock which interconnect existing fractures.

6.1.1 Previous Research on Slope Stability

Several studies on rock slope stability have been performed using stochastically generated fracture patterns and applying simplified or idealized mechanical models for fracturing of intact rock.

Glynn (1979) generated a parallel fracture pattern with the two dimensional Veneziano model (Veneziano, 1979) and used an idealized mechanical model. Specifically, stress increments both in the horizontal and vertical direction were created so that an en echelon fracturing could be reproduced. O'Reilly (1980) modified Glynn's idealized mechanical model by introducing both tensile and shear failure mechanisms. A further step was accomplished by Shair (1981) who expanded O'Reilly's geometric model by considering two fracture sets which are not necessarily parallel. Einstein, et al. (1983) summarized the above mentioned models and performed parametric studies on geometric and on mechanical variables.

All these models are restricted to two dimensions by assuming an infinite extension in the third direction. A solution to the three dimensional stability problems using stochastic fracture geometry was described by O'Reilly (1980) ; however, his application was resticted to a special case. Numerous applications dealing with three dimensional sliding failure and deterministic geomet ist (e.g., Einstein, et al., 1979 ; Hoek and Bray, 1981), but they are not directly related to the problem at hand.

6.1.2 Proposed Work on Slope Stability Analysis

In this preliminary study, we will first consider the fully persistent model of a slope. The fully persistent model is useful when rock block behavior dominates slope stability. Later, fracturing through intact rock and between existing fractures will be included. For simplicity, our work will be confined to two dimensional case only. Slope stability modelling will be approached as follows :

1. Geometric construction of fracture paths : Depending on the actual fracture pattern, either the homogeneous Poisson point (or fiber) process model or the hierarchical fiber process model is used to generate the network of fractures in a given two dimensional slope.
2. Kinematic updating : All the possible rock blocks (i.e., fully persistent blocks) that can be produced with the fracture pattern are simulated and those which are kinematically admissible are identified.
3. Kinetic (or mechanical) testing : Using an appropriate mechanical model, the kinetic stability of the kinematically movable blocks is determined.
4. Parametric Studies : By changing values of both the mechanical and the geometric parameters, the significance and the influence of these variables on overall behavior of the slope stability is studied.

6.2 Generation of Fracture Pattern

Generation of planar fracture patterns in a given rock slope is basically the same as that in a two dimensional planar surface. Therefore, various fiber process models can be used (see Chapter 4). Conventional models (e.g., the two dimensional Baecher model and the Dershowitz model, see Dershowitz and Einstein, 1988) used the homogeneous Poisson process model for fracture locations and an exponential or lognormal distribution of fracture lengths. Very often, in slope stability problems, fracture traces are assumed to be parallel to each other. However, in this study, a modified orientation distribution is also used; either the von Mises or the wrapped normal orientation distribution is adopted, in addition to the parallel orientation distribution (As we will discuss later, the non-parallel orientation will cause a more complicated rock block behavior. That is, the generated rock blocks consisting of one or two sets of parallel fractures have a well defined prescribed face on which sliding occurs, whereas the blocks constructed from the non-parallel orientations do not have unique sliding faces. The sliding faces can only be determined after the simulation of the particular fracture pattern is performed. Also, the kinematic admissibility becomes much more complicated to check). Finally, one can include the fracture termination probability as in Dershowitz and Einstein (1988) or by using our new concept discussed in Chapter 5. At present, this is not done here.

Once the slope boundaries and fracture network are generated, one needs to find the effective fractures. An *effective fracture* is defined as a fracture which can be a part (i.e., a face) of a rock block. It must, therefore, have at least two intersection points with other fractures or with the slope boundaries. In essence, *effective fractures* are interconnected fractures. Effective fractures are thus relevant both in slope stability problems and in flow through rock masses. One can imagine a few algorithms to find the effective fractures; we will use an iterative searching scheme with which the non-effective fractures are sequentially eliminated during iterations. As shown in Fig. 6-1, in each iteration, the non-effective fractures are eliminated, and this in turn induces further eliminations of the fractures which are originally connected to them. Our results show that, within at most 5

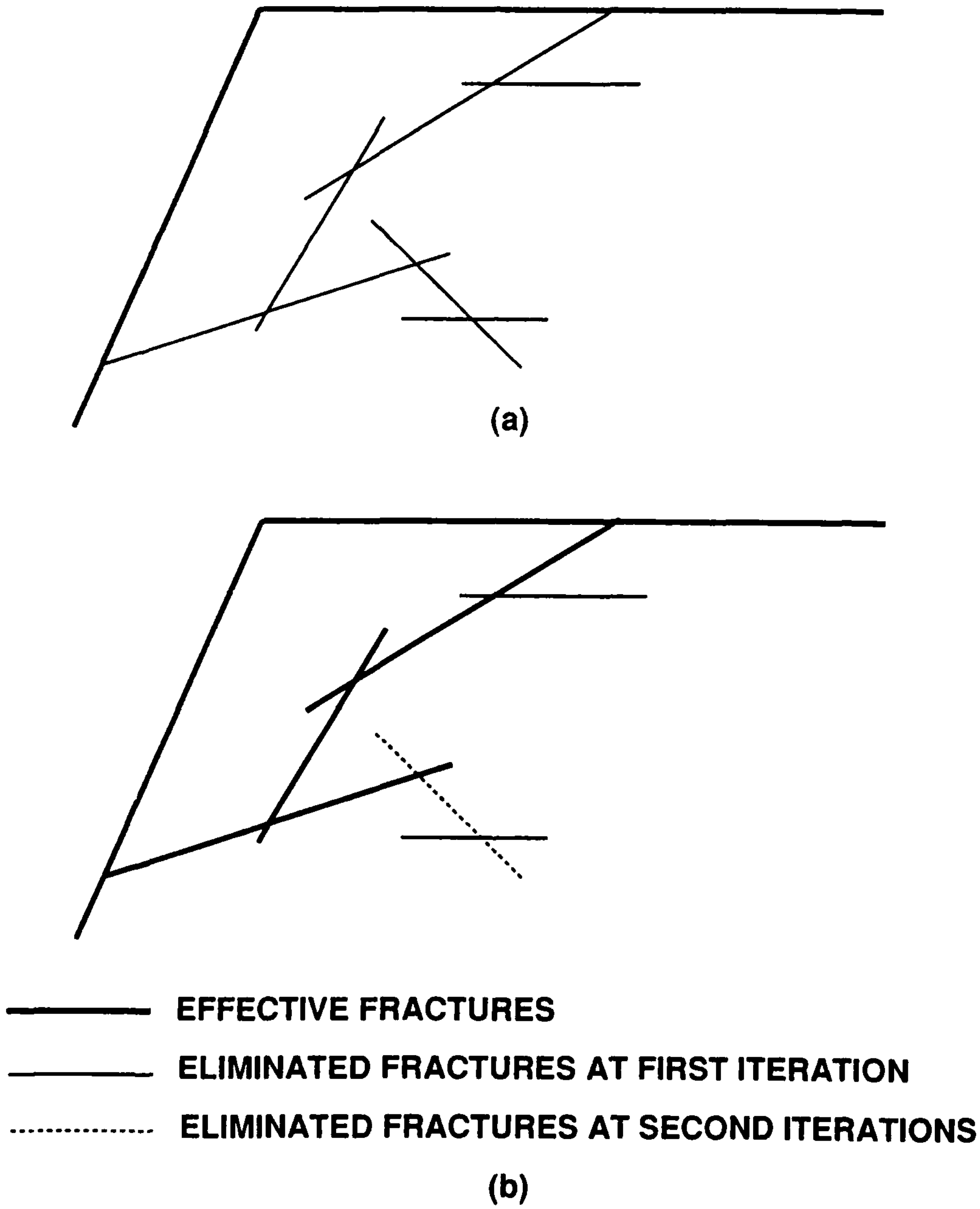

(a)
EFFECTIVE FRACTURES
ELIMINATED FRACTURES AT FIRST ITERATION
ELIMINATED FRACTURES AT SECOND ITERATIONS
(b)

Figure 6-1: Iterative Searching Scheme for Effective Fractures
(a) Initial Fracture Network
(b) Iterative Searching Scheme for Effective Fractures

iterations, all the non-effective fractures are eliminated. The elimination of the non-effective fractures is also an important step in relation to the program storage design since the effective fractures rather than total number of fractures or intersection points are used to define kinematically admissible fracture paths. As a consequence, one can improve the efficiency of the path-searching algorithm.

6.3 Kinematic Analysis

After removing the non-effective fractures and constructing all possible rock blocks, one needs to check whether a constructed fracture path is kinematically admissible or not. Since the fracture orientations (i.e., dip angles) are not necessarily parallel, one cannot predict sliding faces or the block shapes beforehand. Therefore, a systematic kinematic analysis is necessary.

6.3.1 Construction of Connectivity Matrices

A connectivity matix is used to find all the appropriate fracture paths (or fully persistent rock blocks) among effective fractures.

Assume that the intersection points between effective fractures and slope boundaries are known. One can, then, construct a connectivity matrix by employing effective fractures and slope boundaries (i.e., slope face and free surface, see Fig. 6-2). Since all the fracture paths are composed of either face-to-face paths (paths starting and ending at the slope face) or face-to-surface paths (paths starting in slope face and ending in surface), one can consider the slope boundaries as effective fractures. For this reason, the corresponding dimension of the connectivity matrix, N, is

$$N = Number\ of\ Effective\ fractures + 2 \tag{6.1}$$

To construct a connectivity matrix, we assume that each element of the matrix represents either the x- or the y-coordinate of the intersection point and each row or column of the matrix represents an effective fracture or a slope boundary. Therefore, the connectivity matrix becomes a square matrix with dimension *N*. To simplify the searching algorithm, one can assume that the first row/column is the slope face and the last row/column is the surface of the slope. One can then store the intersection points in the appropriate places of the connectivity matrix according to their relationship to the effective fractures or slope boundaries. Since an effective fracture and a slope boundary cannot intersect itself, the connectivity matrix has a zero diagonal and is symmetric. An example connectivity matrix is shown in Fig. 6-3. In Fig. 6-3 (a), effective fractures including slope boundaries are numbered such that the slope face is effective fracture 1; the surface is effective fracture 5. The remaining fractures are numbered arbitrarily. Fig. 6-3 (b) is the complete matrix for this case. For example, effective fracture 1 intersects effective fracture 2 at x-coordinate 1, and effective fracture 3 intersects effective fractures 2 as well as 4 at x-coordinates 2 and 3, respectively.

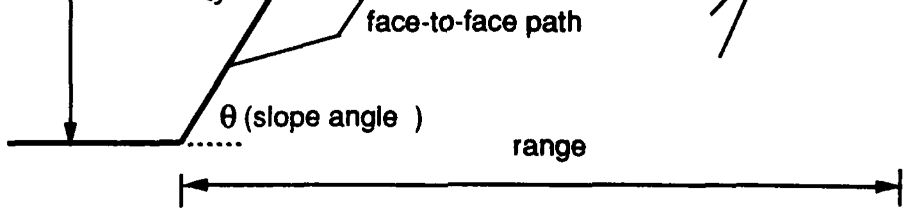
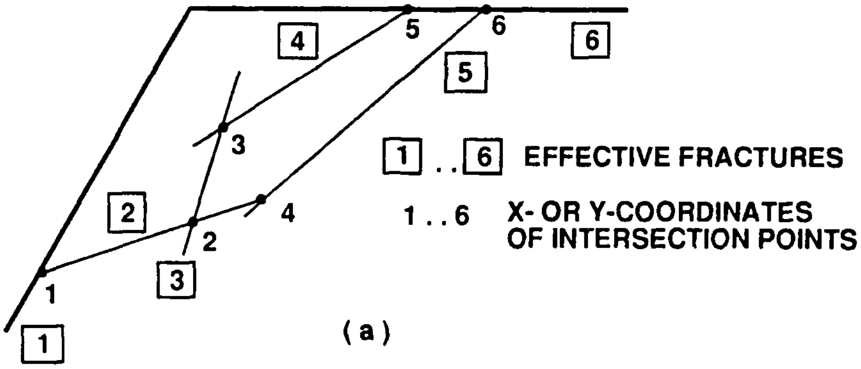


Figure 6-2: Typical Rock Slope Configuration


4
5
6
6
5
3
2
4
2
3
1
1
1 .. 6 EFFECTIVE FRACTURES
1 . . 6 X- OR Y-COORDINATES OF INTERSECTION POINTS

(a)

$$[C_x] = \begin{array}{c|cccccc} & \boxed{1} & \boxed{2} & \boxed{3} & \boxed{4} & \boxed{5} & \boxed{6} \\ \hline \boxed{1} & 0 & 1 & 0 & 0 & 0 & 0 \\ \boxed{2} & 1 & 0 & 2 & 0 & 4 & 0 \\ \boxed{3} & 0 & 2 & 0 & 3 & 0 & 0 \\ \boxed{4} & 0 & 0 & 3 & 0 & 0 & 5 \\ \boxed{5} & 0 & 4 & 0 & 0 & 0 & 6 \\ \boxed{6} & 0 & 0 & 0 & 5 & 6 & 0 \end{array}$$

(b)

STEPS	ROW #	COLUMN #	COORDINATE
STEP 1	$\boxed{1}$		
STEP 2	$\boxed{1}$	$\boxed{2}$	1
STEP 3	$\boxed{2}$	$\boxed{1}$	1
STEP 4	$\boxed{2}$	$\boxed{3}$	2
STEP 5	$\boxed{3}$	$\boxed{4}$	3
STEP 5	$\boxed{4}$	$\boxed{6}$	5
STEP 5	$\boxed{2}$	$\boxed{5}$	4
STEP 5	$\boxed{5}$	$\boxed{6}$	6

(c)

Figure 6-3: Example of a Connectivity Matrix $[C_x]$
(a) A Simulated Effective Fracture Pattern
(b) Construction of a Connectivity Matrix (x-coordinate)
(c) Searching Scheme for a Fracture Path

6.3.2 Kinematic Consideration

Once the connectivity matrix constructed, the next step is to find the kinematically possible fracture paths using the connectivity matrix. The information on the fracture paths are the coordinates of intersection points which become vertices of the blocks. For this, either the x- or y-coordinate system can be employed in the connectivity matrix and in the searching scheme. The other x- or y-coordinates of the intersection points which are not used in the connectivity matrix can be found easily if the intersection points are numbered sequentially, and the coordinates of the intersection points are stored correspondingly.

The searching scheme for the kinematically admissible rock blocks is similar to the dynamic programming scheme used by Glynn (1979). It searches for a fracture path and kinematic admissibility of the path simultaneously. In the following, we will explain these separately for reasons of clarity:

The search for a fracture path which defines a fully persistent rock block follows the sequence of steps cited below (see also Fig. 6-3 (c)):

1. Start always at row 1 of the connectivity matrix, where row 1 represents the slope face.
2. Search for the non-zero element in row 1. If the non-zero element is encountered in the *j*-th column, store the information (i.e., the corresponding effective fracture and intersection coordinate) and go to row *j*. It means that an effective fracture, designated as the *j*-th fracture, is intersecting the slope face at the stored value of the connectivity matrix.
3. Starting from column 1 of fracture *j*, find the non-zero elements of the *j*-th row of the matrix. If the non-zero element is the same as the one stored before, neglect it since this is an interesection point which was previously found. If the column 1 has a non-zero element, store the information. It represents a face-to-face path. If the last column has a non-zero element, store it also. In this case, the corresponding fracture path will be a face-to-surface path (Recall that row/column 1 represents a slope face and row/column *N* represents a free surface, where *N* is the total number of the effective fractures including slope boundaries, see Eq. (6.1)).
4. In step 3, if the *k*-th column of row *j* has a non-zero element, go to row *k*, i.e., fracture *k*.
5. Iterate steps 3 and 4 for row *k* to find a complete fracture path. Whenever a

fracture path is detected in row *l* and non-zero elements still exist in row *l*, go back to the previous step (i.e., row *l - 1*) and search for another non-zero element (i.e., another fracture path) until no more non-zero elements are detected (see steps 5 of Fig. 6-3(c)).

To eliminate the kinematically inadmissible fracture paths, the following assumptions are made during the searching algorithm :

1. If the relative location of the next intersection point compared to the position of the current intersection point lies in region IV (see, Fig. 6-4a), neglect the corresponding fracture path since it is kinematically inadmissible. Unless this assumption is made, a sliding rock block will penetrate into the rock mass.
2. If row *j* of the connectivity matrix is encountered again during a path searching process, ignore the corresponding path. It means that a concave or convex sub-block within a rock block is generated and, which is kinematically unacceptable unless a breakdown of concave/convex sub-block occurs (Fig. 6-4b).
3. Concerning the face-to-face path, even if kinematically admissible, whenever the location of the starting intersection point of a path is higher (i.e., nearer to the slope surface) than that of the ending point, neglect that path (Fig. 6-4c). In this manner, the face-to-face path will be only counted once, namely going upward.
4. To eliminate a stair-shaped path (Fig. 6-4d), do not consider paths when the the advancing fracture path has negative orientation (see Fig. 6-4a for symbols) and the absolute value of it, $|\beta|$, is greater than the minimum dip angle, $|\alpha|$, on which a rock block is sliding down.

6.4 Kinetic Analysis

In the preceding steps, we have found the kinematically admissible fracture paths. The remaining task is to analyze whether the kinematically admissible fracture paths are kinetically or mechanically stable or not. For this, we will use Coulomb's shear failure criterion as a mechanical model (other models can be easily implemented).

As mentioned in the introduction, we assume that the fracture path making up a rock block is fully persistent and that the dip angles of the fractures are not necessarily the same (i.e., parallel). One cannot, therefore, predict the minimum dip angle on which the rock block is to slide until the kinematic analysis is performed. The minimum dip angle plays an

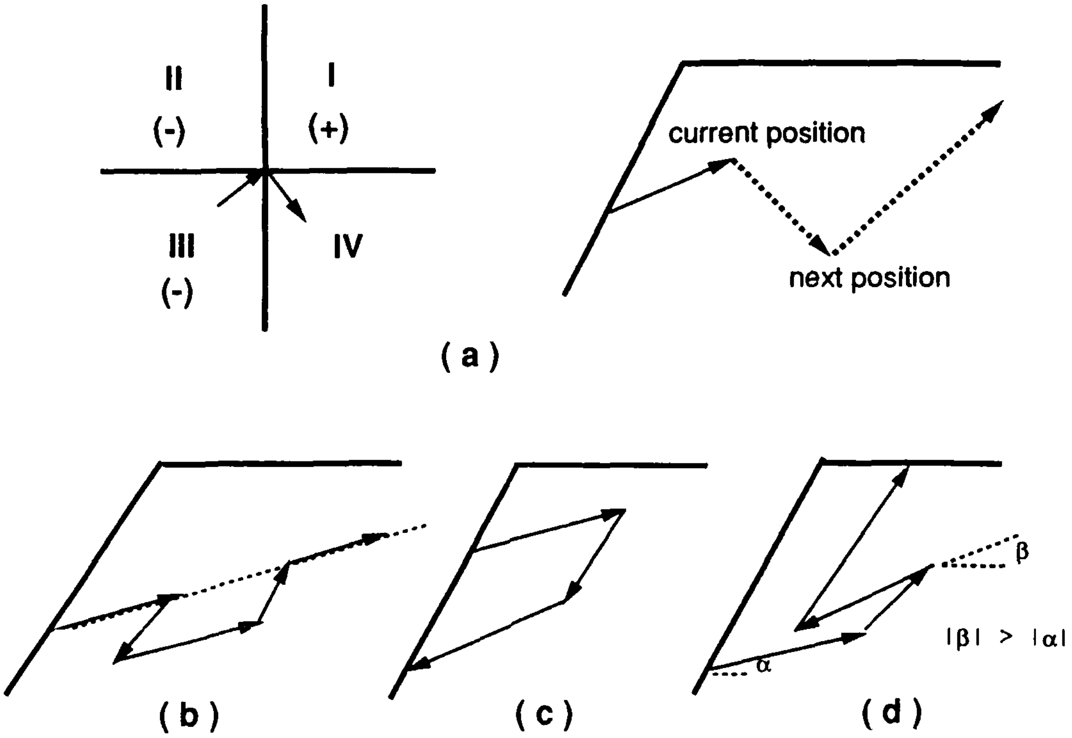


Figure 6-4: Kinematically Inadmissible Fracture Paths

important role when one predicts the block behavior. As shown in Fig. 6-5, if the rock block is assumed to behave as a rigid body, i.e., if an intact rock failure is not allowed, there exists only one contact face on which the block can slide. This is a special case since we have assumed that the fracture orientations are not parallel such that a fracture which has the minimum dip angle becomes the contact (i.e., sliding) face. In the following, we will discuss this particular block behavior in detail.

6.4.1 Assumptions used in Stability Analysis

To solve the slope stability problem, the following assumptions are made in the kinetic analysis :

- The problem is two dimensional, i.e., slope boundaries and fractures are assumed to extend infinitely in the third dimension.

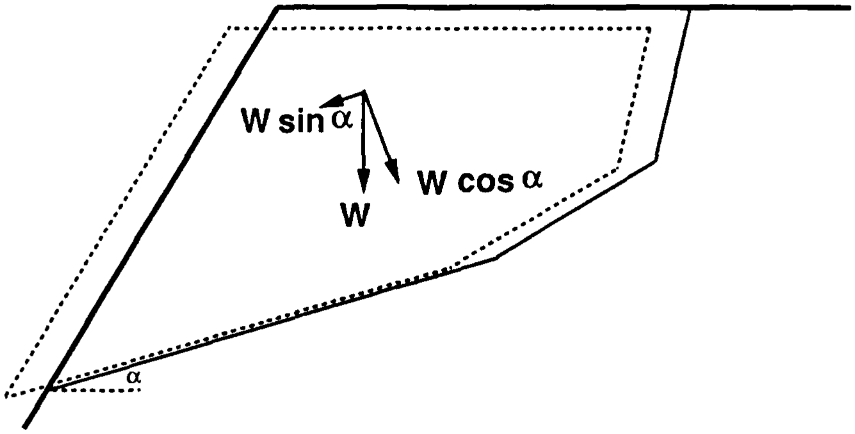


Figure 6-5: Rigid Body Motion of a Typical Rock Block Along a Face of Minimum Dip angle α

- Failure of a rock block is assumed to occur along the fracture with the minimum dip angle. Two failure mechanisms are possible ; sliding and separation along the fracture faces.
- Failure occurs when the driving force exceeds the resisting force and we will apply the Coulomb criterion in this case.
- A tensile cut-off stress is used when considering the separation mechanism.
- Water pressure, rock bolt force and seismic effects are neglected.
- Toppling is assumed to be impossible, and, therefore, rotational behavior is not considered.
- A rock block is assumed to be fully persistent and rigid such that an intact rock failure or an en echelon failure mode (i.e, failure of intact rock which connects the adjacent existing fractures) is not possible.

These assumptions are idealized representation of field conditions and future work will eliminate some of these assumptions.

6.4.2 Simplified Mechanical Model for Sliding

As explained above, sliding will only be considered on the fracture plane with the minimum dip angle (one may include the tensile cut-off stress, (σ_{oj}), when considering separation between blocks). As is well known, the shearing resistance on a sliding fracture face is

$$\tau_j = C_j + \sigma_n \tan \phi_j \tag{6.2}$$

and correspondingly the resisting force, R_j, is

$$R_j = \tau_j A \tag{6.3}$$

where, σ_n is the average normal stress which can be derived from the total block weight W ; C_j and ϕ_j are the cohesion and the friction angle of a fractured face, respectively ; A is the initial contact area of the fracture with the minimum dip angle. Since a two dimensional plane failure case is considered, the rock block is assumed to have a unit length in the third dimension.

The resisting force due to separation can also be included such that the total resisting forces on a rock block can consist of the shearing force on a sliding face and the tensile cut-off force on separating faces. The tensile cut-off force, R_{oj}, which is conservatively a scalar sum of all the components along the separating faces can be written as

$$R_{oj} = \sum_{i=1}^{N} \sigma_{oj} A_i \tag{6.4}$$

where, N and A_i are the total number of separation faces and the area of the separating face of a rock block, respectively. The total resisting force is therefore,

$$R = R_j + R_{oj} \tag{6.5}$$

The resulting safety margin, SM (Einstein, et al., 1983), and the safety factor, SF, are calculated as

$$SM = D - R$$

$$SF = \frac{R}{D} \tag{6.6}$$

where, D is the total driving force and can be calculated from Fig. 6-5 as $D = W \sin \alpha$ along with the minimum dip angle α.

6.4.3 Program TRACESIM (see Appendix D for details)

To generate a fracture pattern in a slope, which does not necessarily follow a homogeneous Poisson fracture process, and to perform the kinematic as well as the kinetic analysis of the fully persistent rock block model, a new computer code, TRACESIM, was developed. The main features of the program TRACESIM are :

1. Up to 10 effective fractures can be included in one rock block; at present, a rock block which has more than 10 faces is not considered. However, modification for more complex block patterns which can include more than 10 effective fractures is possible.
2. Both the homogeneous and the non-homogeneous Poisson fracture pattern can be simulated. However, among non-homogeneous fracture patterns, only the Poisson cluster model (Diggle, 1983) can be considered. For more accurate modelling of the fracture pattern, the existing program POINT (see, Chapter 4 and Appendix D for details) which was developed to simulate the non-homogeneous (hierarchical) fracture process model should be employed.
3. An exponential or a lognormal fracture length distribution can be simulated.
4. The von Mises, wrapped normal, uniform and unidirectional orientation distributions can be simulated.
5. Analysis of two or three fracture sets which have different mechanical (i.e., C_j, ϕ_j and σ_{oj}) or geometric (i.e., trace length distribution and orientation distribution) properties is possible.
6. An auxiliary plotting program TRACESIMP is available for post-processing and plotting of the results.
7. It can consider not only the conventional face-to-surface rock block which may be convex or concave, but also face-to-face blocks.

6.5 An Example of Slope Stability Analysis

To test the applicability of the concept developed in Sections 6.3 and 6.4, and of the program TRACESIM, we will analyze the example shown in Fig. 6-6. As illustrated in Figs. 6-7 and 6-8, there are initially 40 fractures in a slope, and subsequently 6 effective fractures with 10 intersection points. A von Mises orientation distribution with a high concentration factor and an exponential trace length distribution are used in Fig. 6-7. As can be detected visually in Fig. 6-8, there exist 3 kinematically possible fracture paths. The

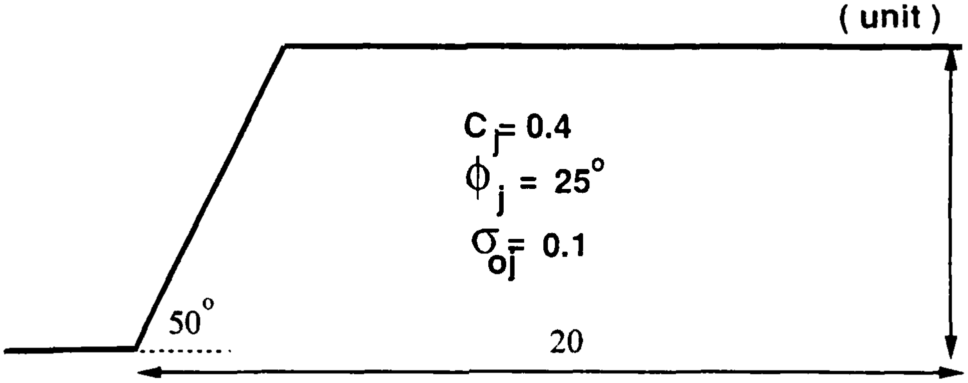


Figure 6-6: Configuration of a Slope Example

kinetic analysis shows that two of the paths are unstable (see, Figs. 6-9 and 6-10). Finally, the safety factors, SF, are calculated for these two unstable paths and the failure path for the case with the minimum SF is plotted in Fig. 6-11.

Through Monte-Carlo simulations, in which one parameter is varied at one time while the others are held constant, one can determine the influence of the geometric and mechanical parameters.

6.6 Parametric Studies of a Slope with Fully Persistent Fractures

So far an example for a fully persistent set of fractures with fixed geometric and mechanical parameters has been given. In this section, parametric studies will be performed to establish relationships between various parameters and the safety factor (i.e., the probability of failure), and to find the most significant parameter. Initially, the mechanical parameters, such as cohesion $\{C_j\}$, friction angle $\{\phi_j\}$ and tensile cut-off stress $\{\sigma_{oj}\}$ of the fractures, are changed one by one. Later, the geometric configurations such as mean orientations, mean trace length and the midpoint model of the fractures are also varied and their influence on the safety factor is investigated.

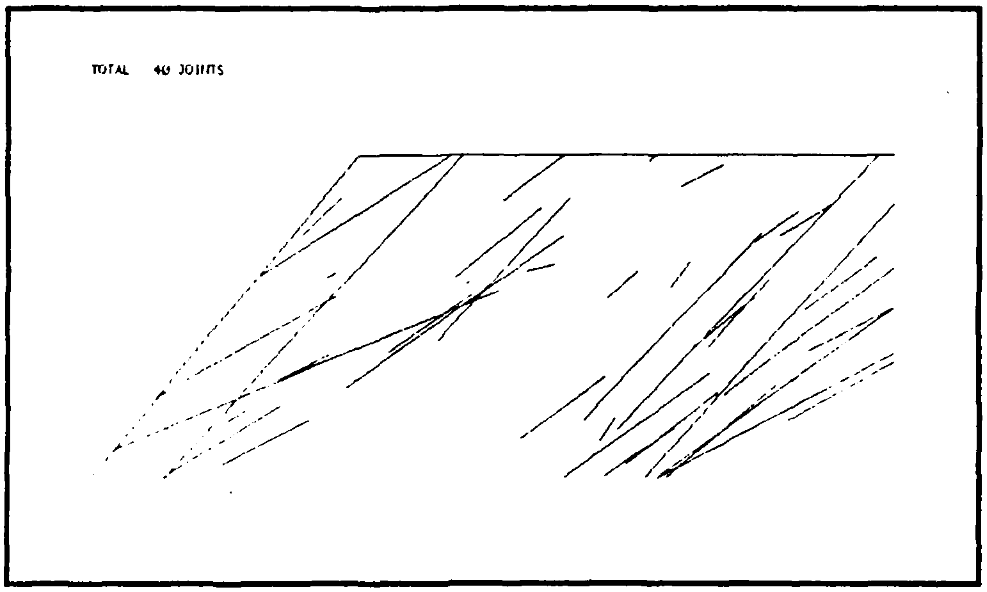


Figure 6-7: Initial Fracture Pattern in a Slope

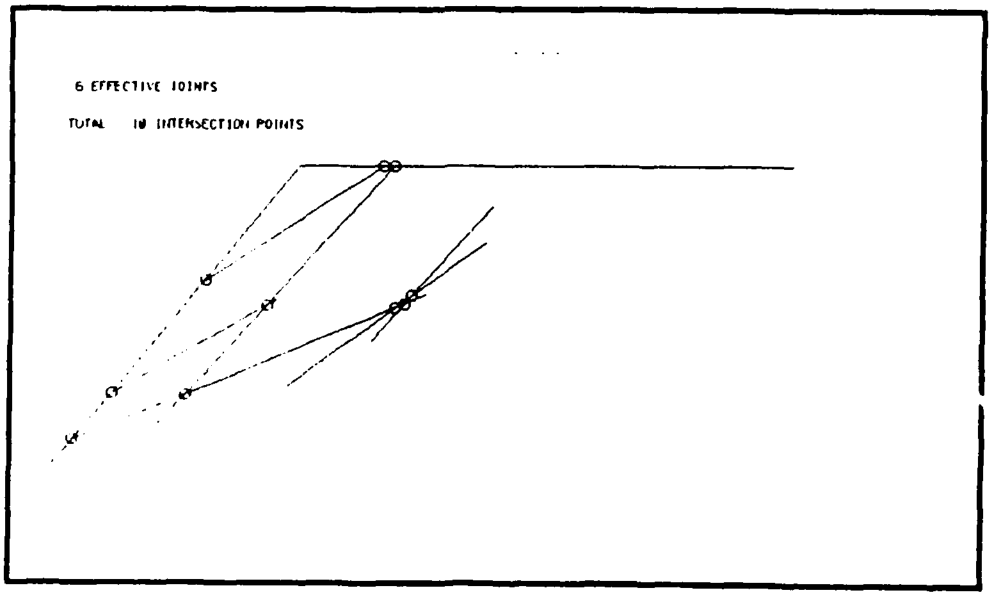


Figure 6-8: Effective Fracture pattern in a Slope

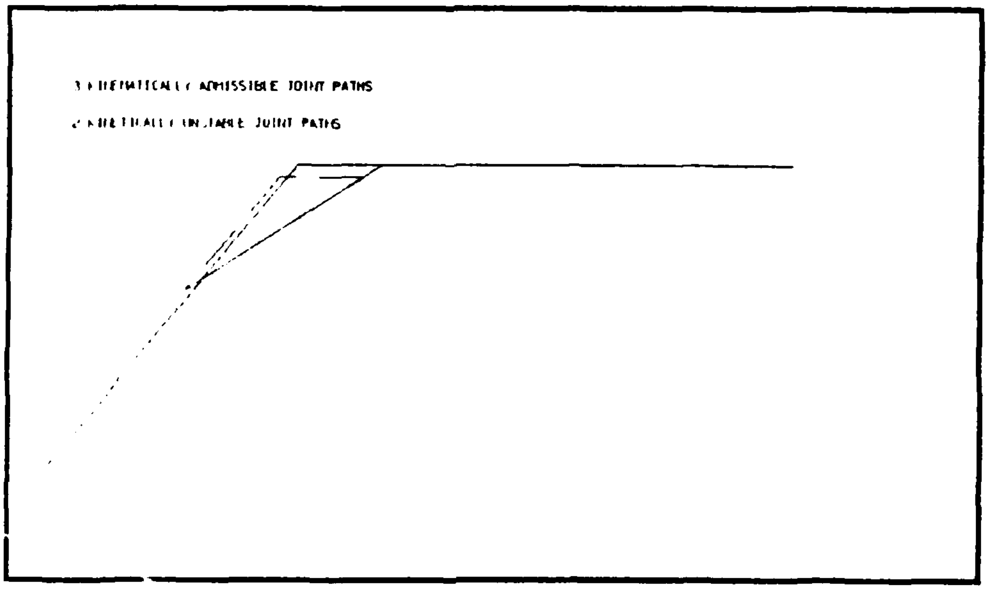

3 KINEMATICALLY ADMISSIBLE JOINT PATHS
2 KINETICALLY UNSTABLE JOINT PATHS

Figure 6-9: One of the Two Unstable Fracture Paths

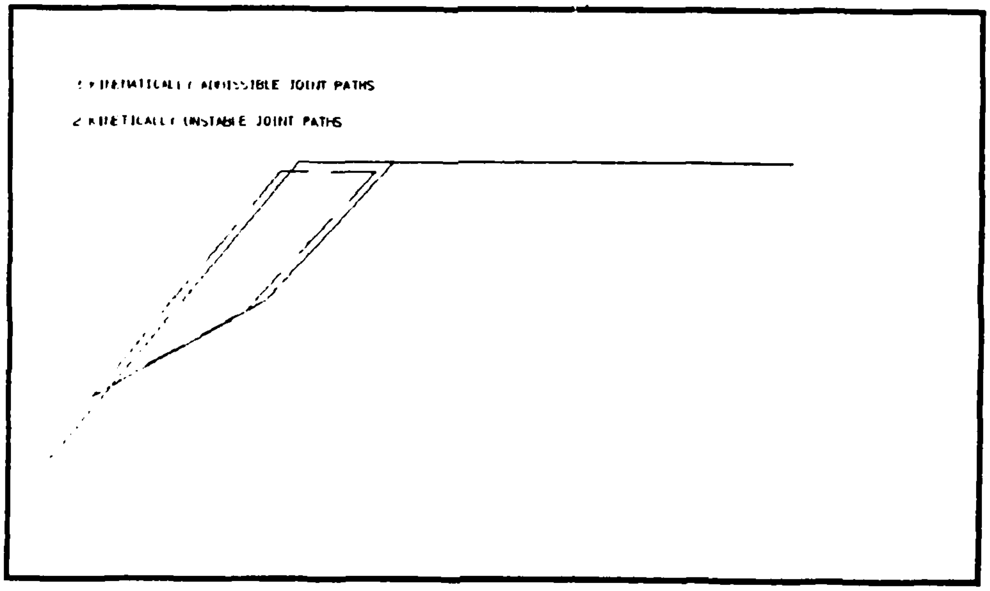

3 KINEMATICALLY ADMISSIBLE JOINT PATHS
2 KINETICALLY UNSTABLE JOINT PATHS

Figure 6-10: One of the Two Unstable Fracture Paths

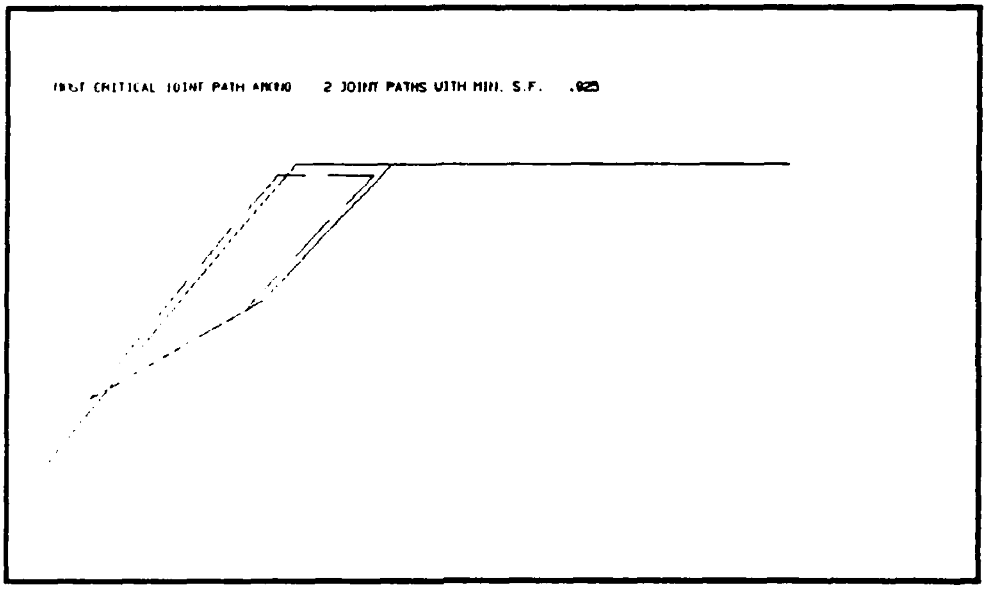

MOST CRITICAL JOINT PATH AMONG 2 JOINT PATHS WITH MIN. S.F. .925

Figure 6-11: Critical Fracture Path

Table 6-1 shows initially fixed and subsequently varied values of all the parameters. As was the case in the previous example, we do not specify the units of the mechanical or geometric parameters. Any consistent system of units can be used. Also, slope configuration is the same as shown in Fig. 6-6 and the initial midpoint model is the homogeneous Poisson process with 50 fractures. Table 6-2 lists the cases which are considered in the present study.

6.6.1 Case 1: Variation of Cohesion

Before proceeding to detailed analyses of the parametric studies, we calculate the probability of failure, P_f. Calculation of P_f is composed of several steps;

1. For a given case, we perform 50 simulations per parameter state of the varied parameter (the parameter states are listed in Tab. 6-1). The resulting data set has hundreds of possible rock blocks, most of which come from the small variations of fracture paths. To simplify the data set, we choose the maximum

Table 6-1 Initial and Changing Values of Parameters

Parameter	Fixed Value	Varied Range	Comments
Model			Homogeneous or Non-homogeneous Poisson Model
No. of Fracture Sets, N_j	1	2	
Cohesion, C_j	1000	0, 500, 1000, 1500	
Friction Angle, ϕ_j	30°	10°, 20°, 30°, 40°	
Tensile Cut-off Stress, σ_{oj}	0	0, 10 50, 100	
Orientation Concentration Factor, κ	20	10, 20, 30, 40	
Mean Fracture Orientation (Dip Angle), α	40°	20°, 30°, 40°, 50°	
Mean Trace Length, m_l	6	4, 6, 8, 10	
Unit Weight of Rock Mass, γ	2200		Fixed

or the minimum volume among rock blocks which have the same sliding angle. We already know that the sliding angle is the minimum dip angle which becomes one of the faces of the fully persistent rock block, and on which the rock block is sliding down. We expect that choosing either the maximum or the minimum volume does not affect the overall properties of the data (e.g., randomness of the number of occurrences of kinematically admissible rock blocks and mean sliding angle. etc) since, even though different in shapes due to path variations, the fracture paths which retain the same sliding angle have essentially the same properties. Fig. 6-12 is a plot of sliding angle v.s. volume of rock blocks for case 1 with parameter state, C_j = 1000 (recall that 50 simulations are run per parameter state). Fig. 6-13 is a simplification of the data in Fig. 6-12, in that only the maximum rock block volume is considered.

Table 6-2 Investigated Cases

Case	Model	N_j	C_j	ϕ_j	σ_{oj}	κ	α	m_l
Case 1	H. P.	1	Varied	Fixed	Fixed	Fixed	Fixed	Fixed
Case 2	H. P.	1	Fixed	Varied	Fixed	Fixed	Fixed	Fixed
Case 3	H. P.	1	Fixed	Fixed	Varied	Fixed	Fixed	Fixed
Case 4	H. P.	1	Fixed	Fixed	Fixed	Varied	Fixed	Fixed
Case 5	H. P.	1	Fixed	Fixed	Fixed	Fixed	Varied	Fixed
Case 6	H. P.	1	Fixed	Fixed	Fixed	Fixed	Fixed	Varied
Case 7	H. P.	2	Fixed	Fixed	Fixed	Fixed	Fixed	Fixed
Case 8	N. P.	1	Fixed	Fixed	Fixed	Fixed	Fixed	Fixed

2. Check the randomness of the number of occurrences of kinematically admissible rock blocks, or kinematic mechanisms. If the kinematically admissible rock blocks occur randomly, i.e., if the number of kinematic mechanisms follow the Poisson distribution during simulations, the corresponding probability of failure, due to the Poisson process properties, can be calculated easily. In the following, we will discuss the procedure to derive the probability of failure in detail. To check the randomness, a χ^2-test is employed. The resulting goodness-of-fit is satisfied at the 10% significance level, see Fig. 6-14. One can, therefore, assume that, during simulations, the kinematically admissible rock blocks occur randomly.

3. Calculate mean ($m_{\alpha|v}$) and standard deviation ($\sigma_{\alpha|v}$) of the sliding angle for a given volume of rock block. The mean sliding angle can be estimated by regression, whereas the standard deviation is calculated from the moving average of the data with respect to the estimated mean sliding angle. We do this because the standard deviation derived from the moving average gives a relatively constant value and, therefore, only one variable, i.e., ($m_{\alpha|v}$), is necessary to calculate the probability of failure for a given volume of rock block. A window width of 10 volume units is used in our case. Fig. 6-15 shows the curve obtained from nonlinear regression which is used to find the mean sliding angle for a given volume of rock block.

4. Calculate the number (or the frequency) of kinematically admissible rock blocks, $\lambda(V)$, as a function of the volume of rock blocks. In our case, either an Exponential or a Gamma distribution can be fitted to the data. However, to prevent the fitted frequencies from approaching zero as the volume of rock block increases, and thus to make it possible to calculate the probability of failure of a very large volume, a Gamma distribution is used. Fig. 6-16 is an

example of the fitting the data with the Gamma distribution to find the frequency of kinematic mechanisms.

5. To include the cohesion effects on a sliding face of the rock block, one needs to calculate the contact lengths for given volumes of rock blocks. In our case, since the contact lengths vary widely and fluctuate inconsistently for a given volume of rock block, they cannot be estimated directly, nor precisely (see, Fig. 6-17). However, one can derive a distribution form for the contact length indirectly as follows:

   - Assume a reference point (see, Fig. 6-17). This reference point can be connected with any of the data points in Fig. 6-17. The reference point, however, does not connect with every data point, e.g., if the volume is too small, one can ignore the corresponding data points since they do not affect the overall relationship between the contact length and the volume of rock block. This is evident when we consider the following steps where the relationship between length and volume is represented by a distribution form.
   - Calculate slopes of lines which connect the reference point and the data points. Also, calculate the maximum and the minimum slope (Fig. 6-17).
   - A Beta distribution is, then, fitted to the slope data to estimate the distribution of the slopes for given volume of rock blocks, Fig. 6-18. In Fig. 6-18, ε is calculated as:

$$\varepsilon = \alpha_{max.} - \alpha \tag{6.7}$$

   where, $\alpha_{max.}$ is the maximum slope found in Fig. 6-17, and α is a currently considered slope. Fitting the data to a Beta distribution is performed with the method of moments (the first and the second moment are used to find the parameters of a Beta distribution).

   - Finally, relate the contact length and the volume of rock block since one of the points (i.e., the reference point) and the distribution of the slopes are now known.

Assuming the slope angle is distributed normally for a given volume of rock block, the probability of failure for a given volume of rock block, $P_{f|v}$, is calculated as:

$$\begin{aligned} P_{f|v} &= P(\{\alpha > \alpha_{crit}\} \mid V) \\ &= 1 - P(\{\alpha < \alpha_{crit}\} \mid V) \\ &= 1 - \int_{-\infty}^{\alpha_{crit}} \frac{1}{\sqrt{2\pi}\,\sigma_{\alpha|v}}\, exp\,[-\frac{1}{2\sigma^2_{\alpha|v}}(\alpha - m_{\alpha|v})^2\,]d\alpha \end{aligned} \tag{6.8}$$

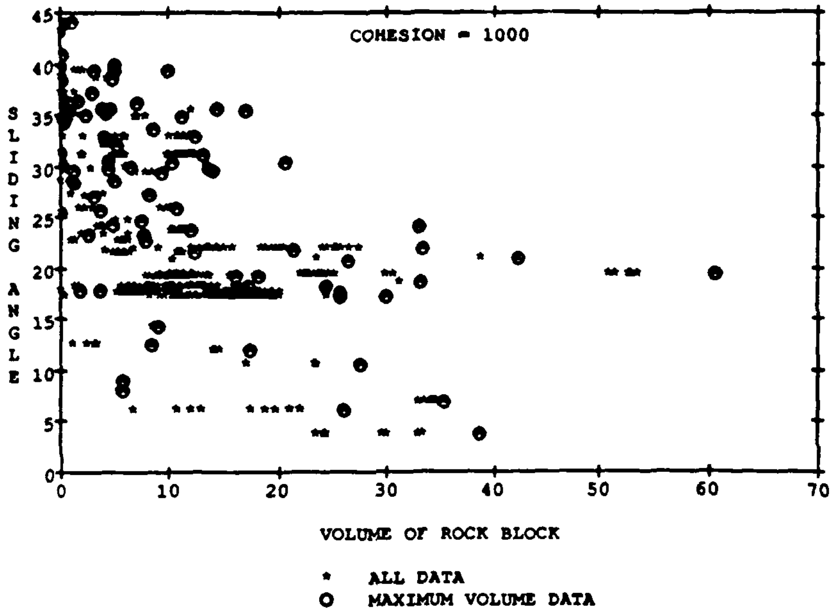


Figure 6-12: A Plot of Sliding Angle v.s. Volume of Rock Block with All Data Set

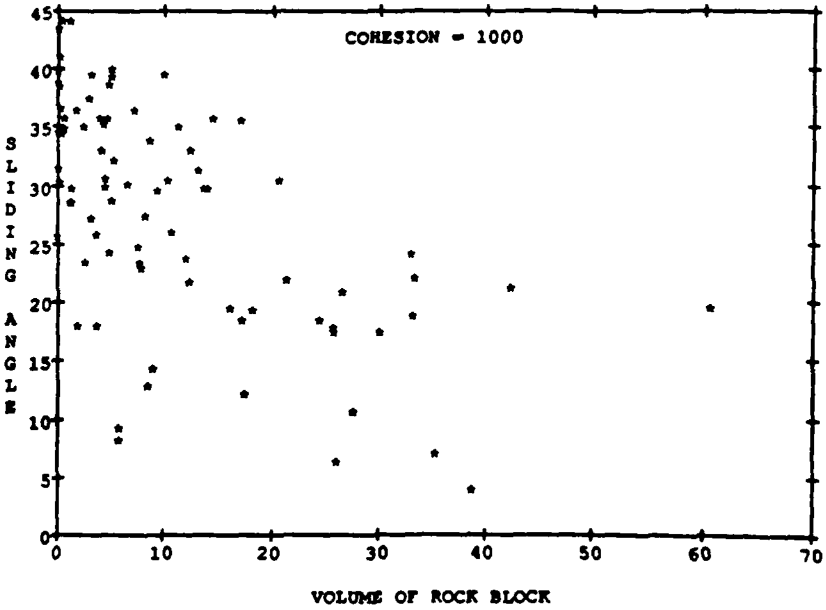


Figure 6-13: Simplification of Fig. 6-12 with Maximum Volume

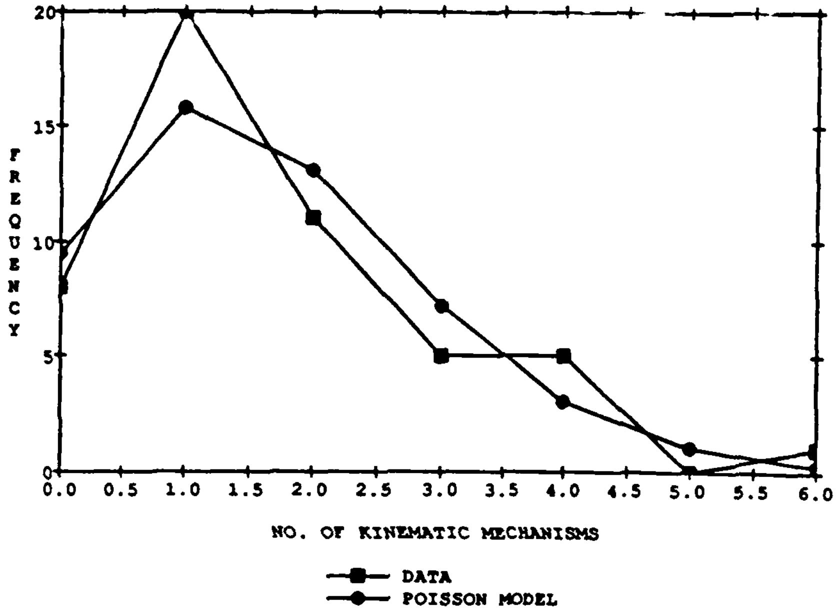


Figure 6-14: Randomness Test of Occurrences of Kinematically Admissible Rock blocks

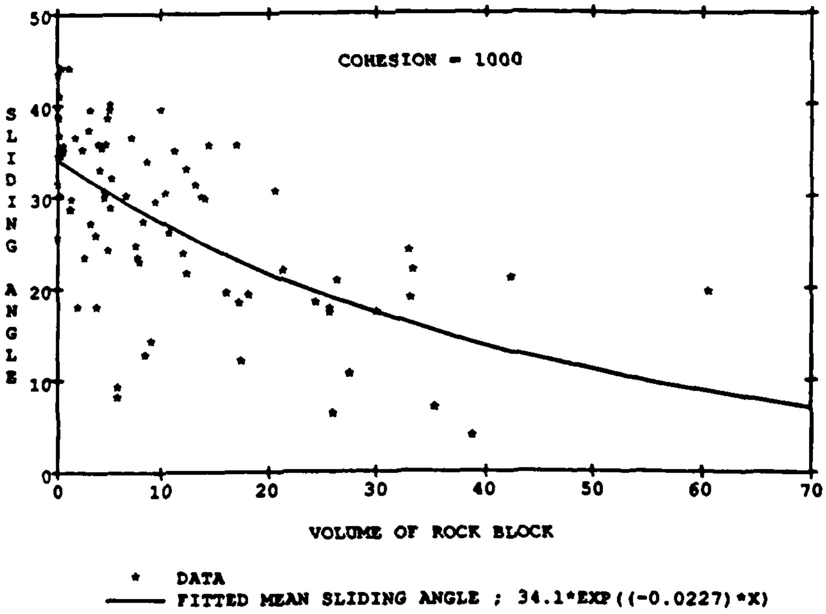


Figure 6-15: Calculation of Mean Sliding Angle Given Volume of Rock Block

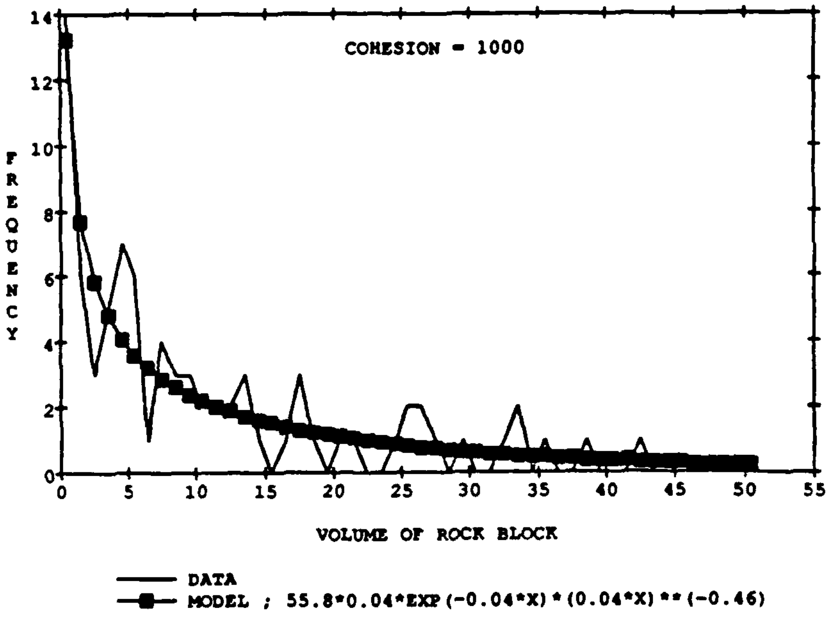


Figure 6-16: Calculation of No. of Kinematic Mechanisms

where, α_{crit} is a critical angle for which the safety factor, S. F., becomes 1. $m_{\alpha|v}$ and $\sigma_{\alpha|v}$ are the mean and the standard deviation of the sliding angle for a given volume of rock block, respectively.

From Step 4 above and from Eq. (6.8), one can estimate the rate of failure occurrences (i.e., the expected number of kinetically unstable rock blocks) within any volume interval; specifically, the rate of failure occurrences within a volume interval $[V_1, V_2]$ can be calculated as:

$$E\,[\,No.\,Failures\,in\,(\,V_1,\;V_2\,)\,] = \int_{V_1}^{V_2} \lambda(V)\,P_{f|v}\,dV \qquad (6.9)$$

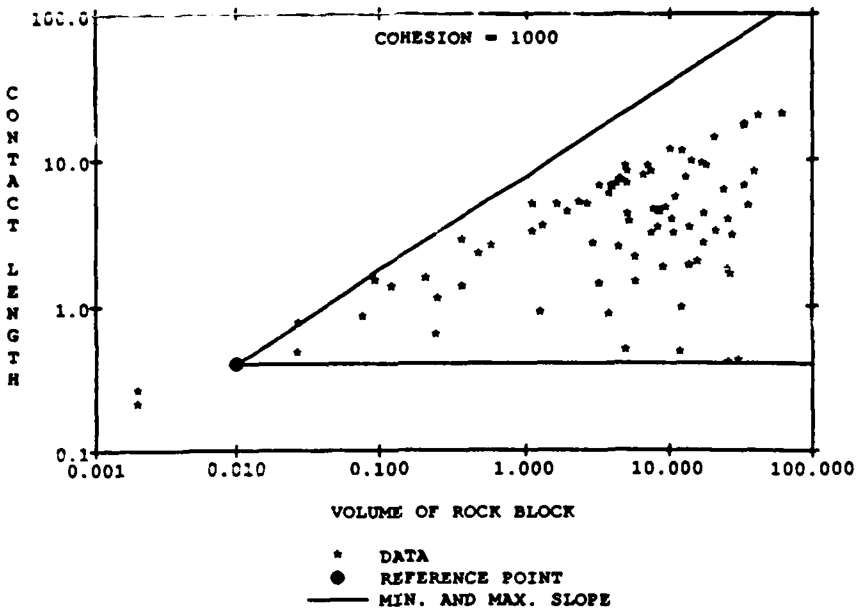


Figure 6-17: Distribution of Contact Length Given Volume of Rock Block

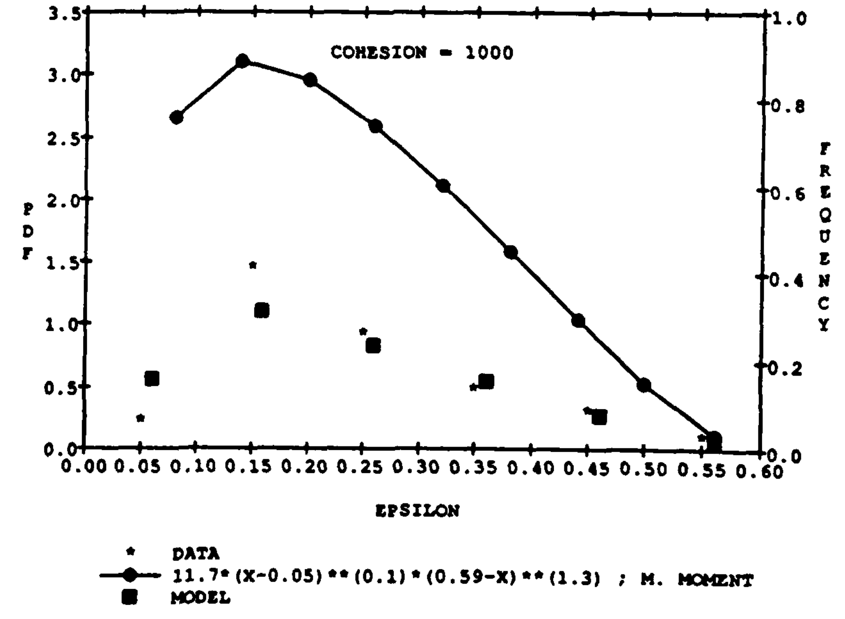


Figure 6-18: Fitting Contact Length to the Beta Distribution

where. $\lambda(V)$ is the frequency of kinematically admissible rock blocks derived in Step 4 in the above. Finally, since the occurrences of kinematically admissible rock blocks are random (see, Step 2 in the above), the probability of failure within a volume interval $[V_1, V_2]$ is estimated from the following equation:

$$P_f(V_1 - V_2) = 1 - e^{-E[No. Failures in (V_1, V_2)]} \qquad (6.10)$$

Up to now, we derived the equation for the probability of failure for a given volume of rock block and, from this, obtained the probability of failure as well as the rate of failure occurrences within a volume interval. Using these relations, one can now investigate the influence of various parameters. We begin with the cohesion effect. To investigate the cohesion effect, the other mechanical and geometric parameters are set constant. As can be imagined, if the cohesion of a fracture is increased from 0 to 1500, the probability of failure for a given volume of rock block significantly decreases, see Fig. 6-19. Note that, when the cohesion has no influence on overall behavior (i.e., $C_j = 0$), the probability of failure will be a monotonically decreasing function, whereas when the cohesion is included in estimating probabilities, one can define a critical or maximum probability of failure which is a function of the volume of rock block. This is an expected result since the slope has a fixed height. It can also explained when one considers the limit state equilibrium. The safety factor for cases affected by friction only will be a function of the sliding angle which is generally inversely proportional to the volume of rock block (see, Fig. 6-15). However, if the cohesion is included in the analysis, the cohesion and the contact length along which cohesion acts will influence the stability. In other words, when the volume is small, the cohesional resistance is more significant than the frictional resistance, while the frictional resistance is dominating the overall behavior of the rock block as the volume of rock block increases. This is because the unit weight of rock mass, γ, is a relatively large value (2200 units) such that, as the volume is increased, its effect on overall behavior (i.e., frictional resistance) greatly increases. One can, therefore, define the critical volume or the maximum probability of failure at the junction of these two effects. Another fact to notice is that, as cohesion increases, the critical volume at which the maximum probability of failure occurs also increases.

The rate of failure occurrences in a volume interval, Eq. (6.9), is plotted in Fig. 6-20, where the interval is set to 1 unit. In Fig. 6-20, one cannot define a maximum rate of failure occurrences with cohesion value 100 (i.e., the maximum rate occurs at 0 volume). This is because the rate of change of the number of kinematic mechanisms in Fig. 6-16 is more rapid than the rate of change of the probability of failure for given small volumes in Fig. 6-19. The probability of failure in a volume interval (1 unit) is depicted in Fig. 6-21.

6.6.2 Case 2: Variation of Friction Angle

When the friction angle increases from 10° to 40°, dramatic changes in overall behavior of the probabilities are induced; specifically the probability of failure for a given volume of rock block is diminished as the friction angle increases, Fig. 6-22. However, the critical volume at which the maximum probability of failure occurs decreases as the friction angle is increased. This is different from the effect of varying cohesion and it can be explained by the fact that, when the friction angle is small, the resisting force due to the friction angle together with the volume (actually, weight) is relatively less significant than the resisting force due to cohesion; whereas, if the friction angle increases, the frictional resistance increases rapidly (recall that the unit weight of rock mass, γ, is a relatively large value, 2200) and it will dominate the overall probabilities. The critical volume is, therefore, decreased.

The rate of failure occurrences and the probability of failure in a volume interval (1 unit) are illustrated in Figs. 6-23 and 6-24, respectively. These two figures clearly show the critical volume interval at which the maximum rate and the maximum probability of failure

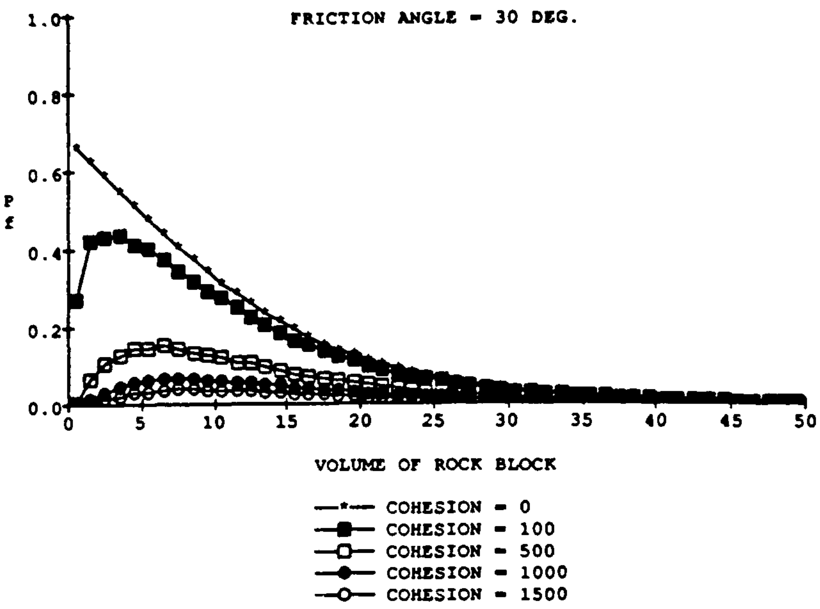


Figure 6-19: P_f Given Volume of Rock Block: Variation of Cohesion

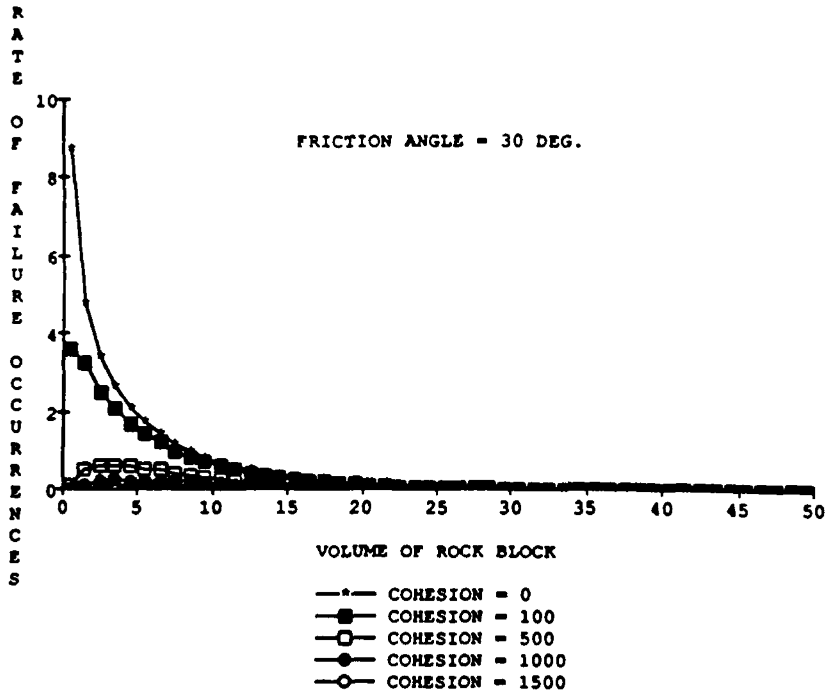


Figure 6-20: Rate of Failure Occurrences in a Volume Interval: Variation of Cohesion

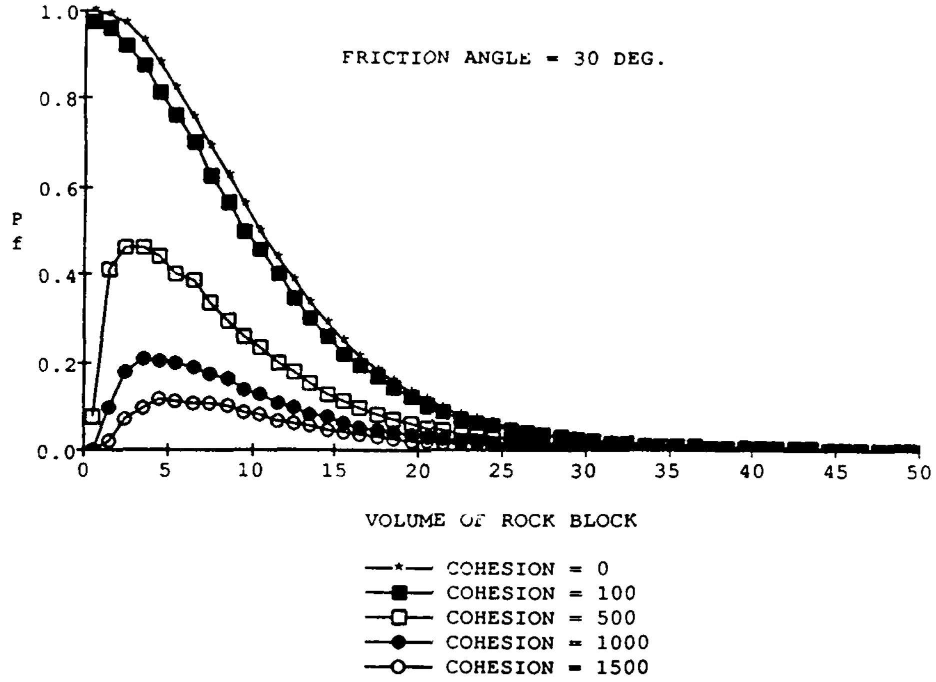


Figure 6-21: P_f in a Volume Interval: Variation of Cohesion

occur. In our slope geometry case, the critical volume interval is roughly in 4 to 5 volume units.

6.6.3 Case 3: Variation of Tensile Cut-off Stress

The tensile cut-off stress is varied between 0 to 100. The probability of failure for a given volume of rock block decreases slightly, Fig. 6-25, as one goes from 0 to 100. One cannot, however, observe any significant changes in overall behavior of the probabilities. This means that, when cohesion is relatively large (i.e., $C_j = 1000$) and the maximum tensile cut-off stress is less than one tenth of the cohesion value, the resisting force due to tne tensile cut-off stress does not dominate the resisting forces and, therefore, the probability of failure.

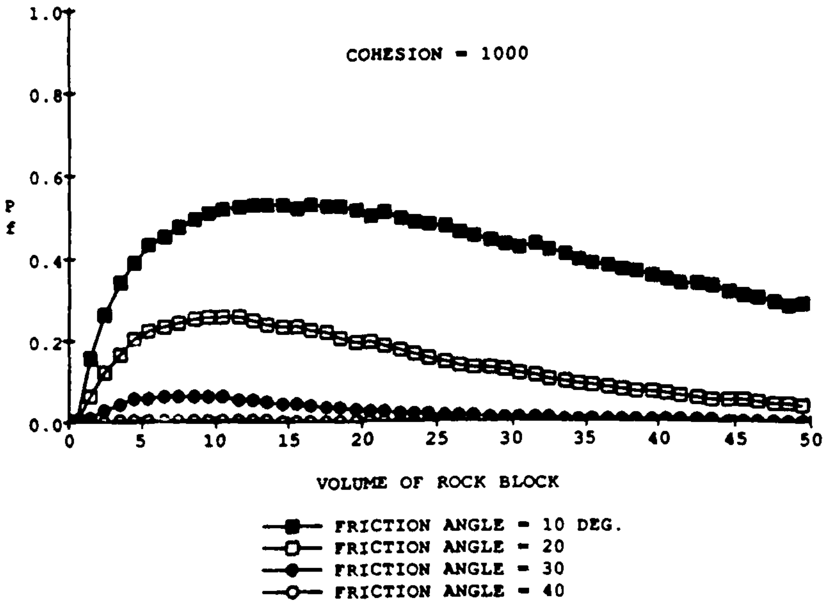


Figure 6-22: P_f Given Volume of Rock Block: Variation of Friction Angle

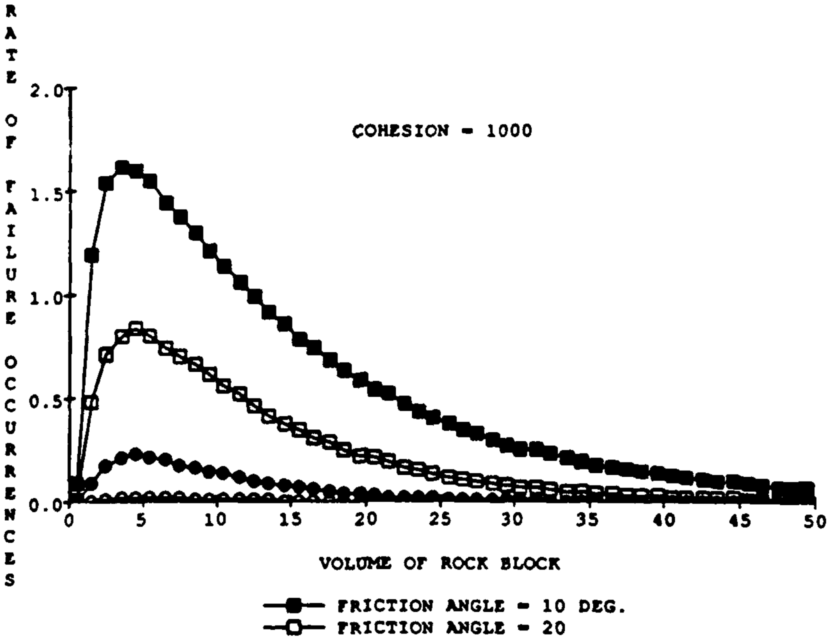


Figure 6-23: Rate of Failure Occurrences
in a Volume Interval: Variation of Friction Angle

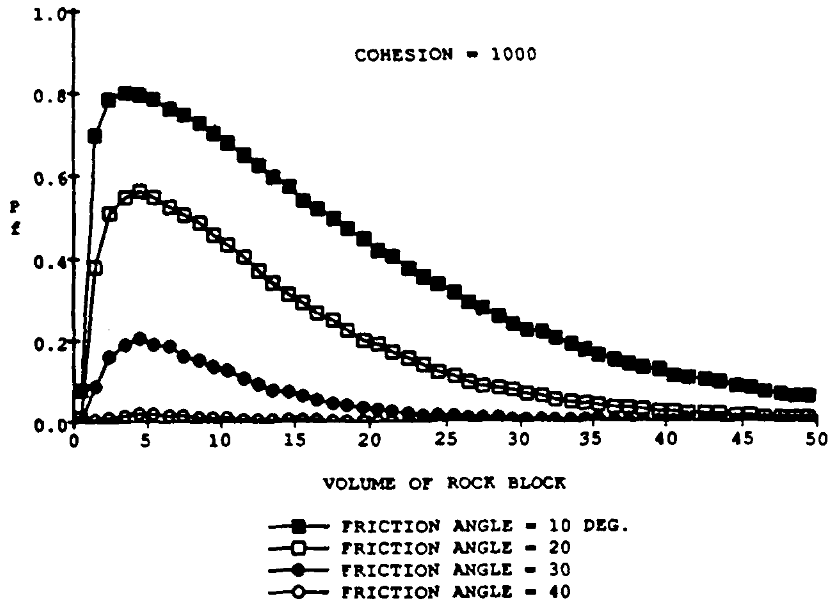


Figure 6-24: P_f in a Volume Interval: Variation of Friction Angle

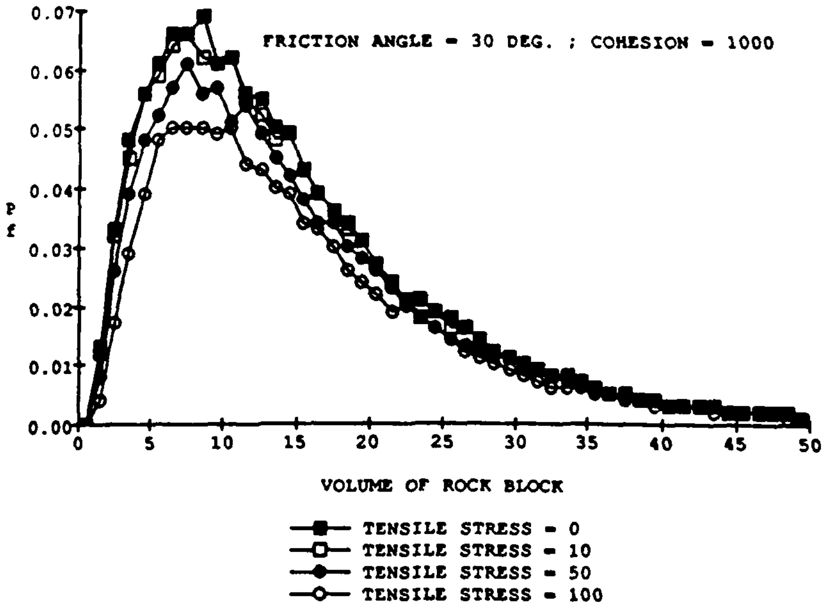


Figure 6-25: P_f Given Volume of Rock Block: Variation of Tensile Cut-off Stress

6.6.4 Case 4: Variation of Concentration Factor of Orientation

Various orientation (dip angle in two dimensional slope stability analysis) models are used when generating a fracture pattern. In our case, we have used the Von Mises distribution with two parameters; one is the mean orientation which is set to 40°, the other is the concentration factor κ which determines the concentration around the mean orientation. If κ is relatively high, the fracture pattern is concentrated (peaked) around the mean orientation; if not, the pattern is dispersed about the mean orientaion and many of kinematically admissible rock blocks can be generated. In the parametric study, concentration factors ranging from 10 up to 40 have been used. Also, to include the influence of cohesion, two different cohesion values are compared.

First, when the cohesion value is relatively large (C_j = 1000), the probability of failure for a given volume of rock block increases with κ up to $\kappa = 20$ because the mean sliding angle becomes steep (recall the mean sliding angle is different from the mean orientation in that the mean sliding angle is the minimum dip angle which constitutes the sliding face of a rock block), see Fig. 6-26. However, as the concentration factor is further increased, the mean sliding angle as well as the contact length of the sliding face increase. (Fig. 6-26). At this point, the resisting force due to cohesion effect dominates the overall behavior and, correspondingly, the probability of failure for a given volume of rock block decreases. Eventually, when the fractures becomes parallel (very high κ), the probability of failure for a given volume of rock block will be zero, Fig. 6-27. Second, if the cohesion value is relatively small (C_j = 100), the probability of failure for a given volume of rock block gradually increases and finally becomes 1 as the pattern becomes parallel, Fig. 6-28. This is because the cohesional resistance is not so significant compared with the frictional resistance of the rock block. From the comparison of these two cases, one can conclude that, when the cohesion is large, the probability of failure for a given volume of rock block increases up to a favorable concentration factor (in our case, $\kappa = 20$) where the contact length (i.e., the cohesion effect) does not contribute to the overall behavior. However, as the fractures approach a parallel pattern (in our case, when κ is 30 or more), the contact lengths are getting longer and finally all the blocks have triangular shapes where the faces of a triangle are composed of two slope boundaries (slope face and surface) and a contact face, Fig. 6-26. It means that the resisting force due to cohesion becomes significant and the probability of failure for a given volume of rock block rapidly decreases and finally becomes zero. When the cohesion effect is negligible, the probability of failure for a given volume of rock block generally increases and finally becomes 1 as the pattern becomes parallel. One more fact to notice is that, when the pattern is parallel, a maximum limit volume exists since our model of a slope has a toe; in the present simulations, the maximum is 11 volume units.

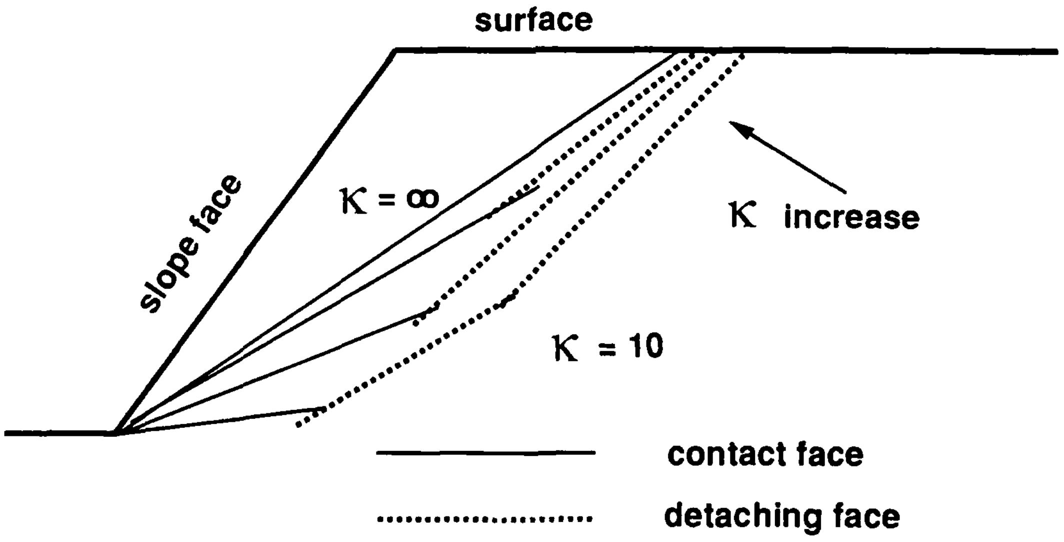


Figure 6-26: Typical Fracture Pattern with Increasing κ

The rate of failure occurrences as well as the probability of failure in a volume interval have the same tendencies as the probability of failure for a given volume of rock block.

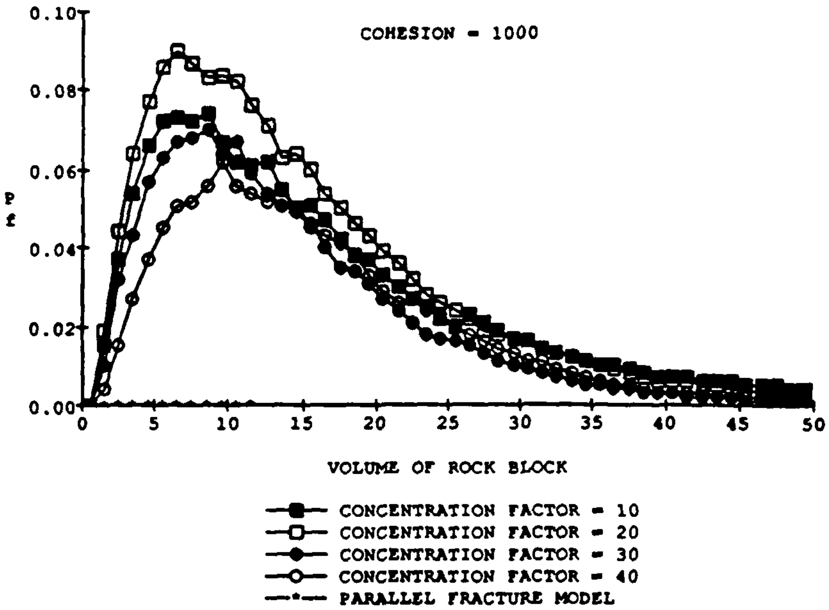


Figure 6-27: P_f Given Volume of Rock Block: Variation of Concentration Factor with Cohesion = 1000

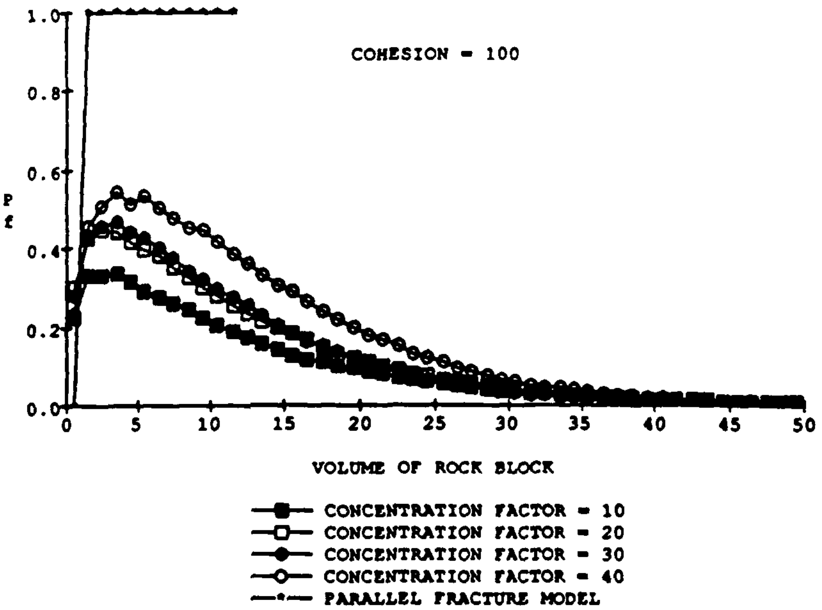


Figure 6-28: P_f Given Volume of Rock Block: Variation of Concentration Factor with Cohesion = 100

6.6.5 Case 5: Variation of Mean Orientation

When the mean orientation (dip angle) of the fractures is changed from 20° to 50°, the overall probability of failure increases. However, since the slope angle, θ, is set to 50° (see Fig. 6-6), the probability of failure for a given volume of rock block sharply decreases if the mean orientation (dip angle) is 50° and if the volume is increased, see Fig. 6-29. This can be explained by the fact that, for a mean orientation 50°, a chance to construct a fully persistent rock block is rarely possible except for small volumes. Figs. 6-30 and 6-31 also show the rate of failure occurrences and the probability of failure in a volume interval.

6.6.6 Case 6: Variation of Mean Trace Length

Using the exponential trace length distribution model and varying the mean trace length from 4 to 10 units, one can find the critical volume and the maximum probability of failure, Fig. 6-32. However, when the mean trace length is less than 4 units, one cannot find any trend since there is little chance to construct a fully persistent rock block. As can be foreseen, when the mean trace length is increased, the probability of failure for a given volume of rock block increases since the mean sliding angle is getting steeper as the mean trace length is increased, see Fig. 6-33. The main reason for the increase in the fitted mean sliding angle is that, since the fracture patterns with different trace length cases are generated with the same random number sequences, the blocks made with large mean trace length include the blocks which were previously made with small mean trace length, see Fig. 6-33. As a result, the fitted mean sliding angle has a possibility to be steeper as the mean trace length is increased. Also, the same behavior occurs both in the rate of failure occurrences and in the probability of failure in a volume interval.

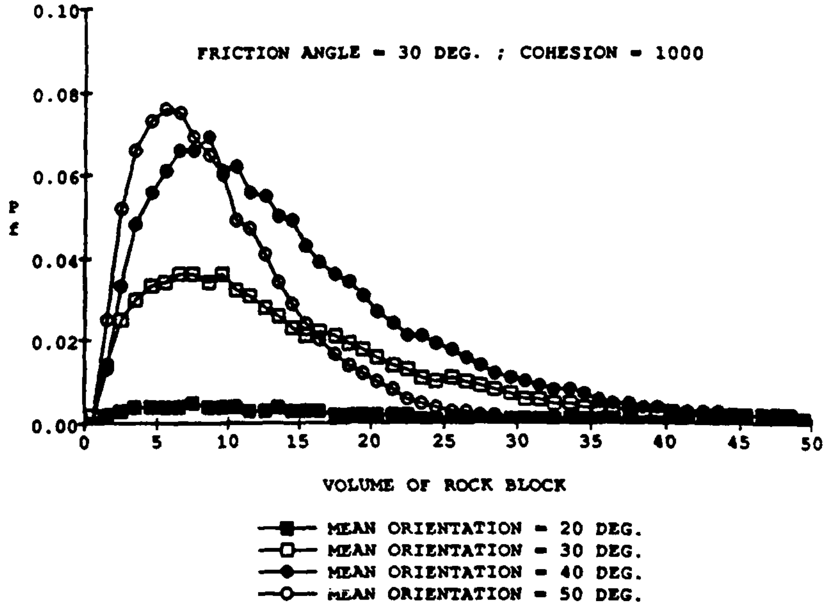


Figure 6-29: P_f Given Volume of Rock Block: Variation of Mean Orientation

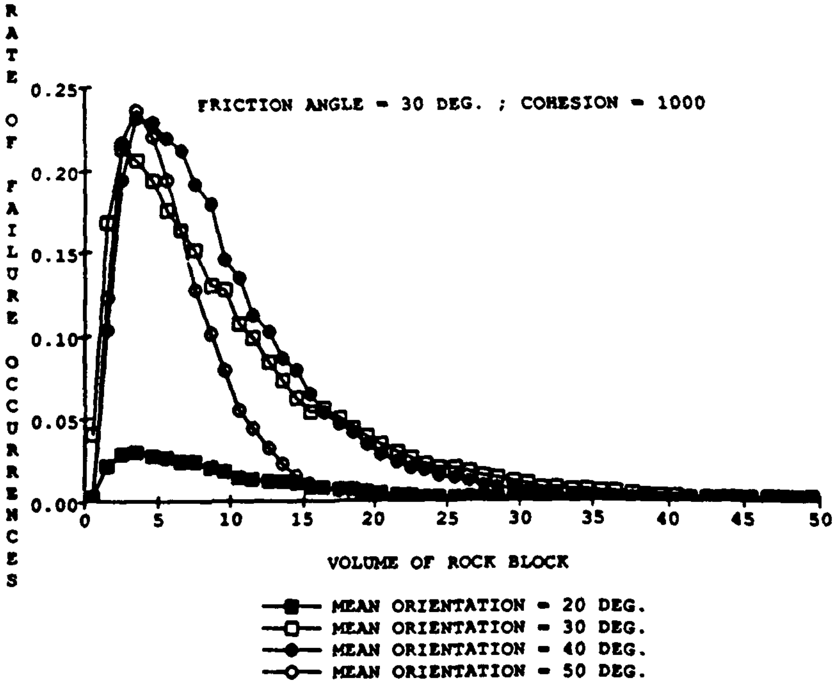


Figure 6-30: Rate of Failure Occurrences in a Volume Interval: Variation of Mean Orientation

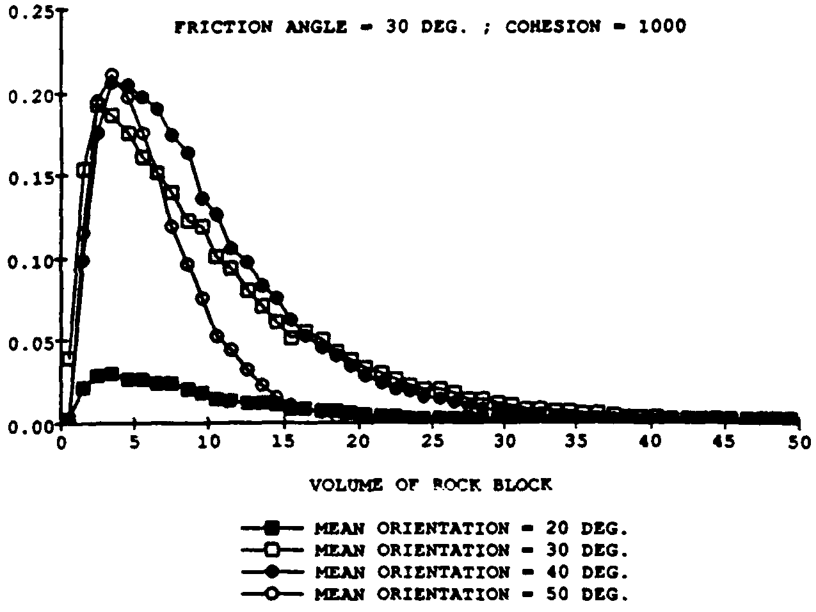


Figure 6-31: P_f in a Volume Interval: Variation of Mean Orientation

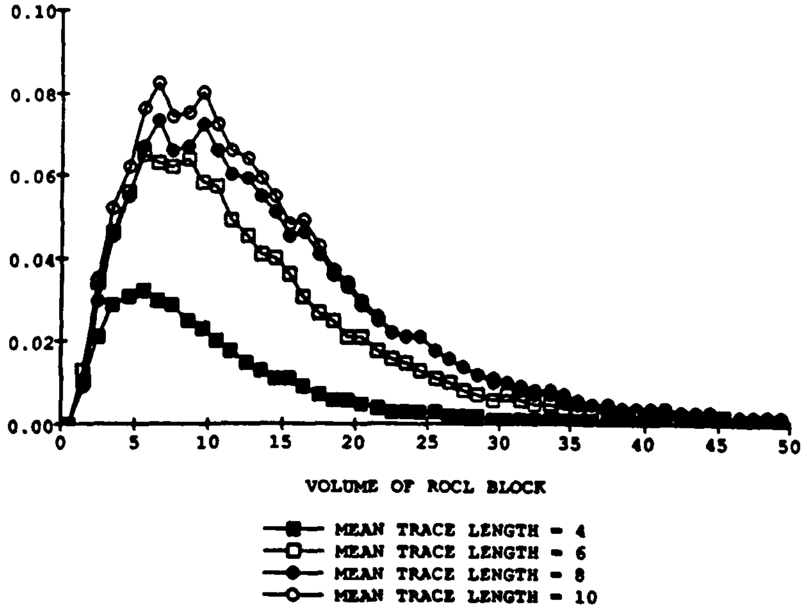


Figure 6-32: P_f in a Volume Interval: Variation of Mean Trace Length

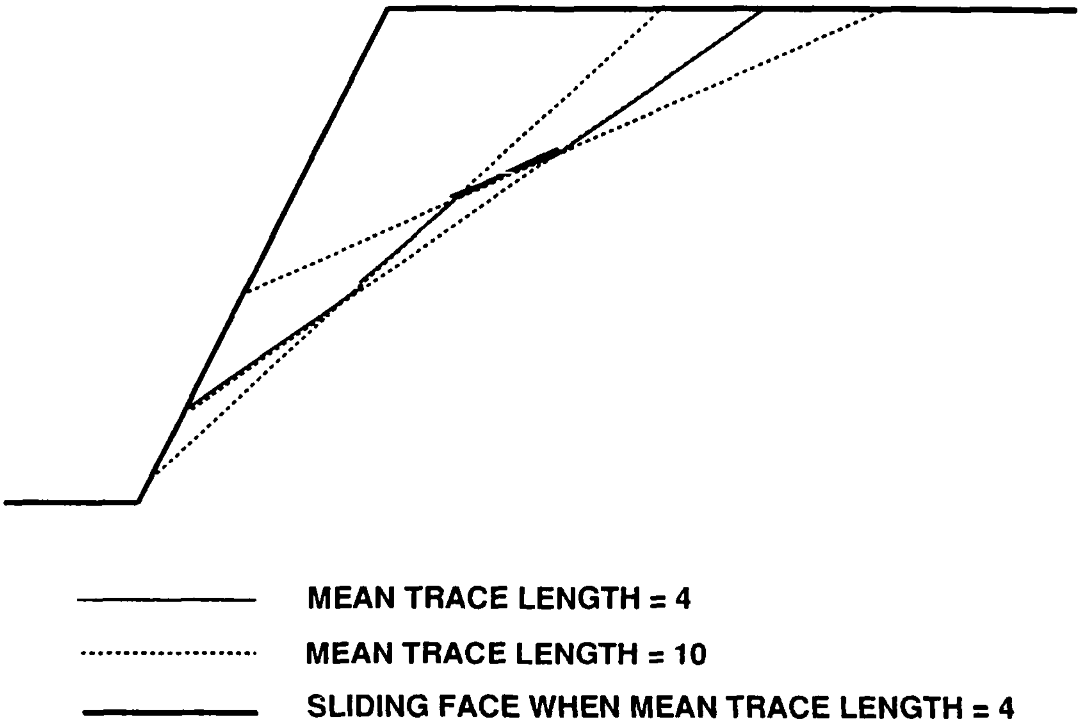


Figure 6-33: Changes of Mean Sliding Angle:
Variation of Mean Trace Length

6.6.7 Case 7: Variation of Number of Fracture Sets

To study the fracture set effect, we divide the fractures into two sets. Set 1 is a set of 25 fractures with mean orientation of 40°; set 2 is a set of 25 fractures with mean orientations ranging from 20° to 130°. Considering the absolutely very low probabilities of failure in Fig. 6-35, one can state that the probability of failure for a given volume of rock block is not greatly influenced by the second set when $C_j = 1000$. This is quite probable since, in the case of 20°, the dip angle on which the rock block is sliding down is dominated by set 2 orientation (i.e., the mean sliding angle decreases, see Fig. 6-34(a)), and therefore, the overall probability of failure is diminished. Also, in the case of set 2 of 40°, combination of two sets determines the sliding dip angle (i.e., the fitted mean sliding angle becomes the steepest if the mean orientations of two sets are equal to 40°, Fig. 6-34(b)) and the probability of failure increases (recall that the sliding dip angle is determined as the minimum dip angle among fractures which constitute the faces of a rock block). However, in the remaining cases, the dominating dip angle on which the block is sliding down is governed by set 1 (i.e., mean sliding angle = 40°) and no significant changes in probabilities are expected, Fig. 6-34(c).

When it comes to the rate of failure occurrences and the probability of failure in a volume interval (Figs. 6-36 and 6-3*1*, respectively), the above mentioned two fracture set cases, i.e., 20° and 40°, as well as the case with 130° have a significant influence on the overall behavior. We may deduce the reason for these results as follows; First, when the mean orientations of two sets are orthogonal (i.e., 40° and 130°), more chances to make fully persistent blocks are expected in small volumes where the mean trace length was fixed at 6 units; however, as the volume is increased, these chances to construct the fully persistent blocks are rarely possible. Second, when the mean orientations of the two sets are 20° and 40°, one has the greatest possibility to construct a fully persistent block except in the small volume region where the orthogonal sets (see above) have greatest probabilities of failure. One can explain this by considering the connectivity matrix concept. That is, if fractures intersect the slope face at many points, more chances to construct the fully persistent rock blocks are possible, and if the mean orientation of fracture set is relatively flat, it will intersect the slope face at many points. Finally, for the same reason as above, when the mean orientations of two sets are equal and are set to 40°, one also has much more chances to make blocks except the second case.

6.6.8 Case 8: Variation of Midpoint Model of Fractures

Up to now, we have utilized the homogeneous Poisson process model when generating the midpoints of fractures. For modelling non-homogeneous Poisson processes, we will employ the Poisson clustering model (rather than the Cox process model) in which the number of seeds (i.e., parents) and the number of daughters are predetermined. In our

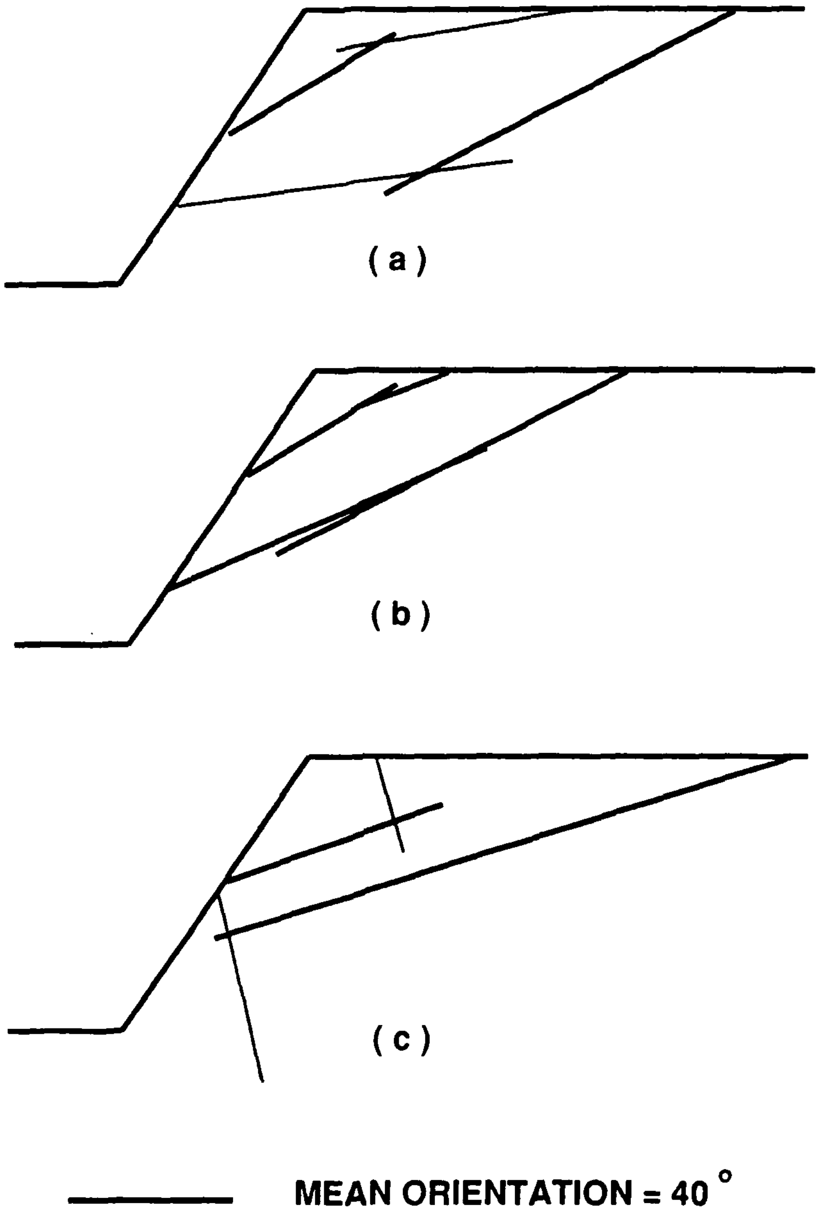


Figure 6-34: Typical Fracture Pattern with Two Sets of Fractures
(a) $\alpha = 20°, 40°$
(b) $\alpha = 40°, 40°$
(c) $\alpha = 130°, 40°$

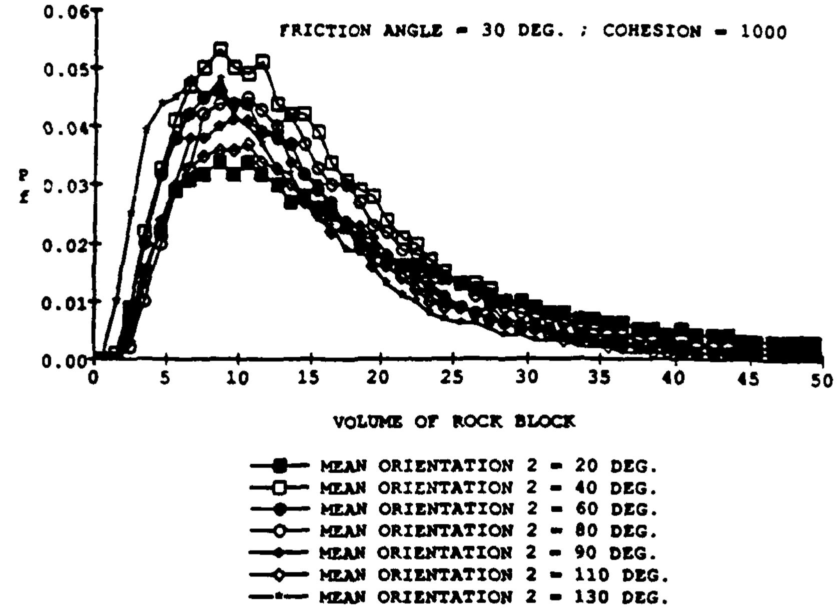


Figure 6-35: P_f Given Volume of Rock Block: Variation of Number of Fracture Sets

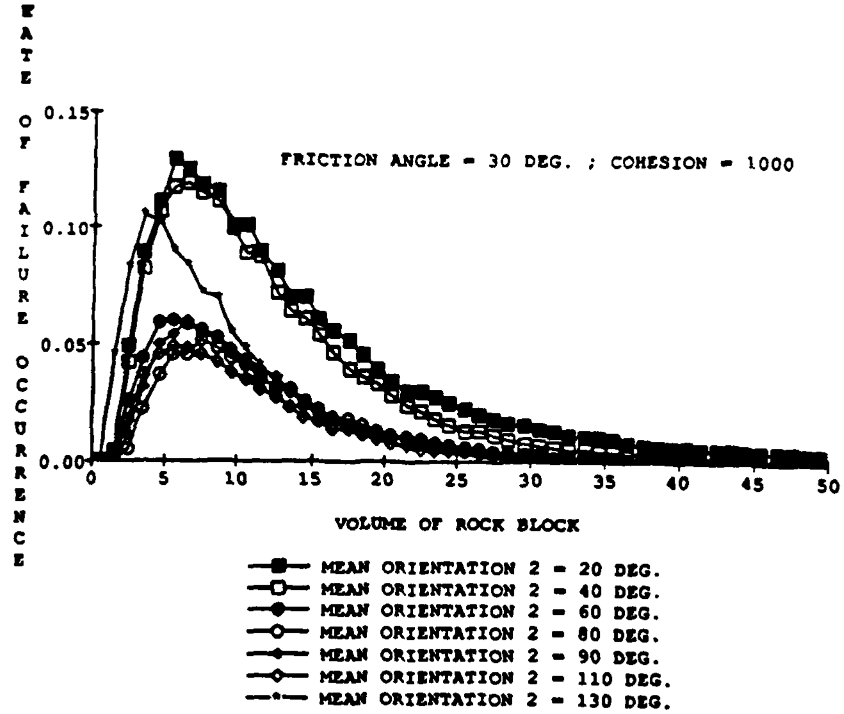


Figure 6-36: Rate of Failure Occurrences in a Volume Interval: Variation of Number of Fracture Sets

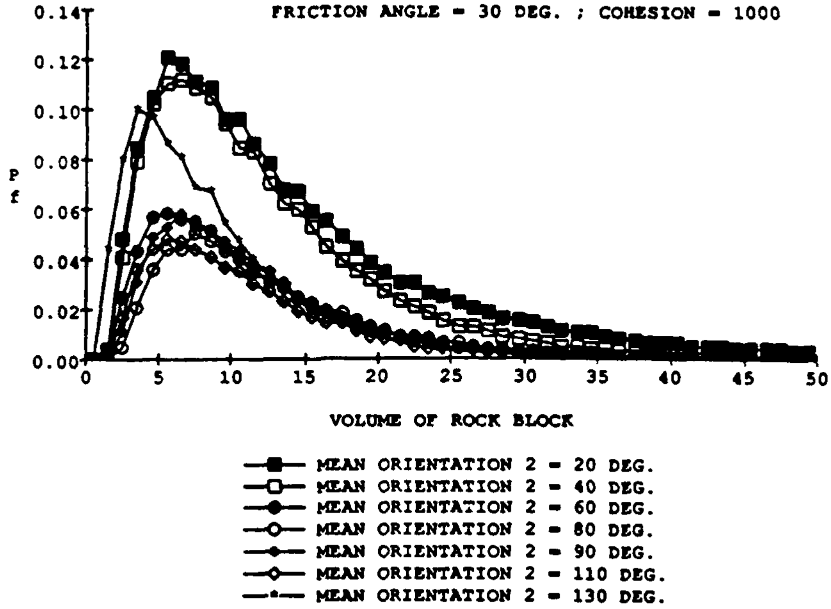


Figure 6-37: P_f in a Volume Interval: Variation of Number of Fracture Sets

case, we will use 3 midpoints as seeds, and the remaining 47 midpoints as daughter fractures. To compare the result, we also use the homogeneous Poisson process model in which 50 fractures are generated radomly over the slope.

The probability of failure for a given volume of rock block increases in the case of non-homogeneous Poisson process model compared to the case of homogeneous Poisson process model since the fractures are clustered around the parent, and therefore, make it easier to construct a fully persistent block. Fig. 6-38 shows the differences in the probability of failure between the non-homogeneous and the homogeneous Poisson process model. Also shown in Figs. 6-39 and 6-40 are the rate of failure occurrences and the probability of failure in a volume interval, respectively.

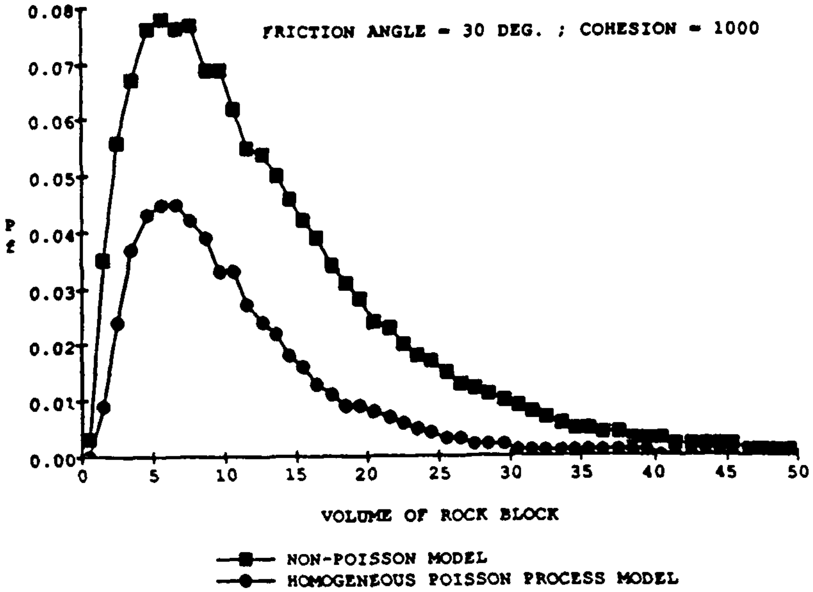


Figure 6-38: P_f Given Volume of Rock Block: Variation of Midpoint Model of Fractures

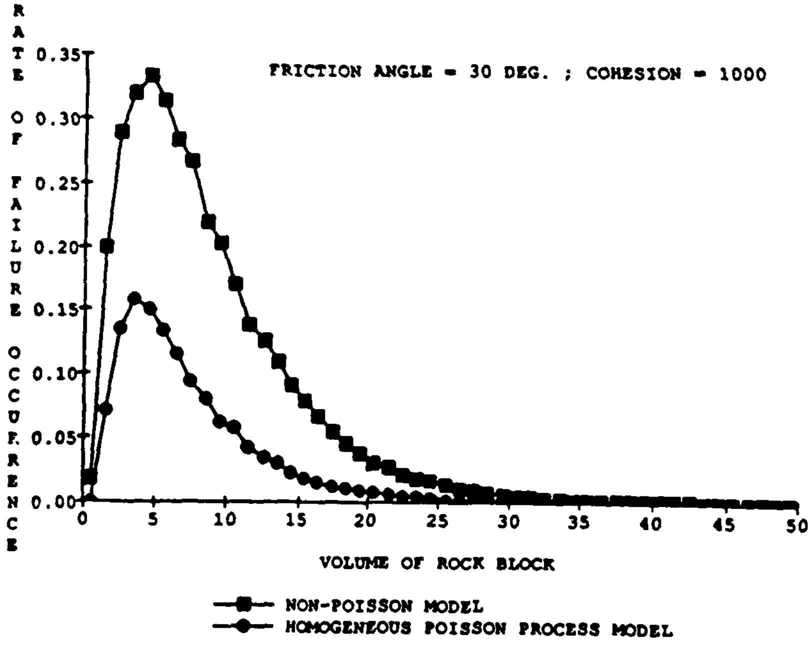


Figure 6-39: Rate of Failure Occurrences in a Volume Interval: Variation of Midpoint Model of Fractures

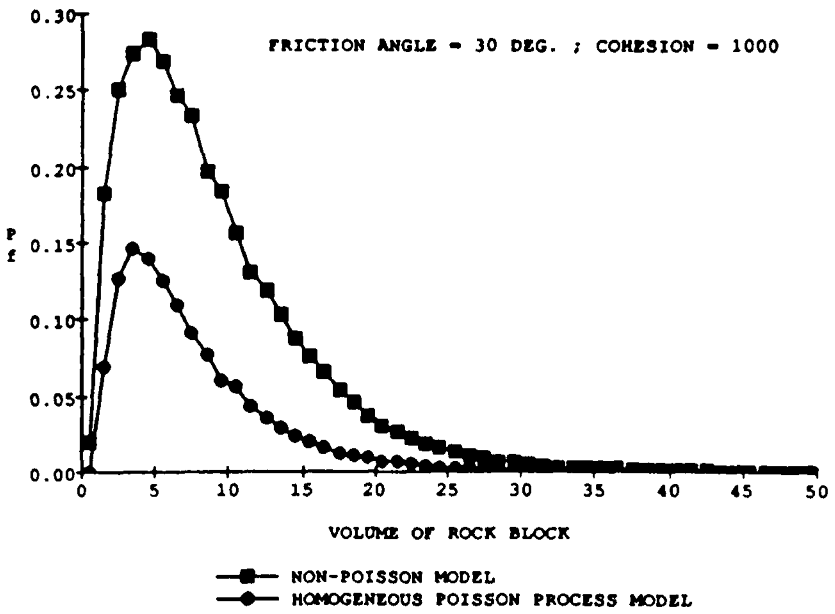


Figure 6-40: P_f in a Volume Interval: Variation of Midpoint Model of Fractures

6.7 Discussion

In this chapter, we investigated blocky rock mass behavior by using stochastically generated fracture patterns and by employing a topological concept. At present, the model is limited to fully persistent fractures, but one can vary mechanical properties of fratures such as cohesion, friction angle and tensile cut-off stress. Also, geometric properties such as mean orientation, concentration factor, mean trace length, number of fracture sets and the midpoint model of fractures can be varied. The effect of varying these parameters, which amounts to a sensitivity study, showed that the number of kinematically admissible rock blocks was largely controlled by the number of effective fractures which intersect the slope face, and that the most important parameter in slope stability analysis was the cohesion as well as the friction angle of a contact face. These two parameters influenced the critical volume (or the maximum probability of failure). Less importantly, the orientation distribution (i.e., mean orientation and concentration factor) affected the overall behavior of the slope stability. However, in our cases, the tensile cut-off stress did not dominate the overall probabilities.

The present limitation of the model in addition to the assumption of persistent fractures is that Program TRACESIM can detect only up to 10 different fractures which become the faces of a rock block. This needs to be modified if the pattern becomes complicated (i.e., more than 10 fracture paths in a rock block). The limitation can be overcome by expanding the connectivity matrix and by adding iteration loops into program TRACESIM. Future research will also make it possible to consider different slope shapes such as slopes with benches and with tension crack developed at the slope surface. Meanwhile, modelling of non-persistent failure requires representation of fracture coalescence.

Chapter 7

SUMMARY, CONCLUSIONS AND OUTLOOK

The main objective of this research was to develop a model and simulation procedure for rock fractures that can reflect the sequential mechanism of fracture formation and, thus, includes dependencies and spatial nonhomogeneities that are often seen in rock outcrops.

Modelling of sequential fracture sets has been accomplished using the hierarchical fracture geometry model. Specifically, the main features of the model are :

1. The spatial variation of trace density is represented by an inhomogeneous Poisson point process or by a doubly stochastic (Cox) point process.
2. A new method, based on maximum likelihood, has been developed for the unbiased estimation of trace length distribution.
3. When several trace sets are present, these sets are analyzed sequentially, according to a hierarchical order. Each set is represented by a conditional stochastic process, conditioning being with regard to the lower-order sets.
4. Two correlation measure methods, the line-kernel function method and the nearest neighbor distance method, have been used to account for the dependencies among fracture sets.
5. Methods from multivariate point processes (Diggle, 1983) have been adapted to the estimation and validation of multivariate fiber processes.
6. A modified termination probability was developed to describe the complex pattern of fracture intersections.

The hierarchical model has been applied to two cases in which detailed fracture patterns have been obtained, one with a single set and one with two sets. The validity of the model was checked by visual comparison between model prediction and mapped pattern and most importantly, by statistical checks such as the second-moment analyses and the Monte Carlo test. A satisfactory fit of the predicted pattern was obtained in both cases.

The hierarchical fracture model has been combined with a topological model to investigate the behavior of a blocky rock mass. The application of the combined hierarchical and topological model proceeds as follows;

1. A stochastic fracture pattern is generated with the hierarchical fracture geometry model.
2. Kinematic as well as kinetic analyses on fully persistent rock blocks are conducted.
3. Finally, using sensitivity analyses, the mechanical and geometric fracture parameters which have the greatest influence are determined.

Slope stability analysis using a fully persistent block model shows that the number of effective fractures which intersect the slope face are the key element for constructing a fully persistent blocky rock mass. Also, in most cases, we could define the critical volume which gives the maximum probability of failure for a given rock block volume.

So far the hierarchical model has been applied to two dimensions only, i.e., to fracture traces in a plane. Further work on the stochastic fracture geometry model will have to be focused on the three dimensional expansion of the existing model. One of the possible three dimensional models can be based on the homogeneous, isotropic Poisson RACS (RAndom Closed Set) process. This idealized model can be expanded to a three dimensional hierarchical model. Clearly, this requires establishing relationships between two dimensional trace length distributions and three dimensional fracture size distributions as well as the consideration of three dimensional orientation distributions.

Concerning topological modelling of a rock mass, two issues will have to be considered in the future work; one is the implementation of a more realistic mechanical model for rock fracturing, notably the mechanism of coalescence of non persistent fractures; the other is an expansion of the two dimensional slope stability problem into a three dimensional one.

It should be noted that, while this research has concentrated on an application to rock slope stability, the basis has been laid for a much broader application. The hierarchical fracture model can be used in any problem involving fractured rock masses. The topological model is similarly useful in other rock mass problem such as flow through fractured rock masses. Most importantly, by expanding the mechanical aspects to include coalescence of existing fractures, the combined hierarchical and topological model can be used in any problem involving solid bodies with discontinuities. Notably, it will be possible to model fractured (cracked) concrete structures. On the other hand, a better understanding of basic small and large scale geologic processes will also be possible.

Appendix A
TESTS FOR RANDOMNESS

Various test methods exist for checking Complete Spatial Randomness. The approaches can be divided into two groups ; one is based on the quadrat count method, while the other uses a distance measure schemes. We will mainly discuss the latter since the quadrat count method, in some cases, cannot satisfactorily describe the point pattern.

A.1 Distance-based [Plotless] Tests

We will discuss the distance-based test assuming the Poisson point process as a null hypothesis H_0. For later use, we define various distances which have their own characteristics. Define an event as a real point pattern such as a tree or mid point of a trace and a point as an artificial point selected randomly. We denote a event-event distance by W, a point-nearest event distance by X, a point-second nearest event by X_2, a distance from an event selected using a random point to the nearest event on a same half plane by Y and, lastly, a distance from an event selected using a random point to the nearest event on the other half plane by Z (see Fig A-1 for details). Here we assume m as the total number of events and ρ as the population intensity. The summation ($\sum$) in the following equations is for $i = 1,..,m$.

Clark and Evans (1954) developed a test based on the inter-event distances. Assuming a point population intensity ρ in a region, the mean observed distance can be represented as

$$\overline{W}_A = \frac{\sum W}{m} \qquad (A.1)$$

The mean distance which would be expected if the pattern is f^llowing CSR,

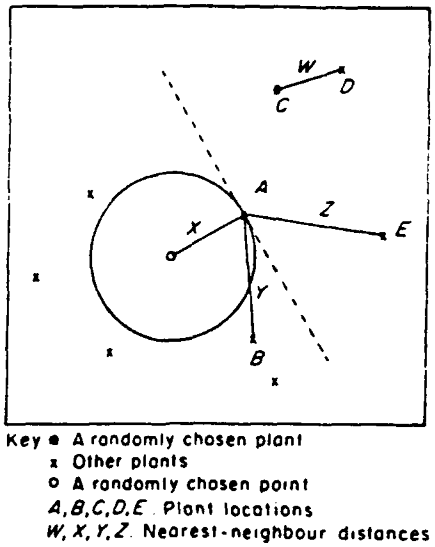

W
C
D
A
Z
E
X
Y
B
Key ● A randomly chosen plant
x Other plants
o A randomly chosen point
A,B,C,D,E Plant locations
W,X,Y,Z Nearest-neighbour distances

Figure A-1: Various Nearest-neighbor Measurements

$$W_E = \frac{1}{2\sqrt{\rho}} \tag{A.2}$$

The ratio

$$h_{CE} = \frac{\overline{W}_A}{W_E} \tag{A.3}$$

can be used as a measure of the degree of departure from CSR. In CSR, $h_{CE} = 1$, under conditions of maximum agregation, h_{CE} becomes 0.

Hopkins (1954) used X and W distances. His test requires complete enumeration of events in the study region. Under CSR,

$$h_H = \frac{\sum X^2}{\sum W^2} \sim F_{2m,2m} \tag{A.4}$$

Here, relatively small or large values of h_H indicate a regular or aggregated pattern, respectively.

Pielou (1959) used an index of non-randomness by calculating

$$h_P = \pi \hat{\rho} \frac{\sum X^2}{m} \sim N(1, \frac{1}{m}) \tag{A.5}$$

In an aggregated population, one would expect a large value of X^2, giving a higher value of h_P. Mountford (1961) considered the sampling errors and corrected Pielou's index.

Holgate (1965) adopted nearest and second nearest distance X, X_2 and suggested a ratio test, based on the sample mean of the ratio

$$h_{HO} = \{ \sum (\frac{X^2}{X_2^2}) \} / m \sim N(\frac{1}{2}, \frac{1}{12m}) \tag{A.6}$$

We reject the H_0 if the observed value of h_{HO} differs from its expectation by more than k times its sampling standard deviation, where k is the appropriate percentage point of the normal distribution. If, instead of the mean of ratios, the ratio of means is considered

$$h'_{HO} = (\frac{\sum X^2}{X_2^2}) \sim F_{2m,2m} \tag{A.7}$$

These test methods are efficient whether a pattern is distributed regularly or not. Here, a regularity means that the points are evenly distributed over a region.

Eberhardt (1967) considered the moment ratio which is easy to calculate

$$h_E = \frac{E(X^2)}{[E(X)]^2} = (\,coefficient\ of\ variation\,)^2 + 1 = \frac{n \sum X^2}{(\sum X)^2} \tag{A.8}$$

where, h_E increases with increasing tendency for aggregation. Since the sampling distributions of the ratio have not been worked out, an exact test of deviation from CSR is not available. But Eqn (A.8) doesn't require a knowledge of density.

Besag & Gleaves (1973) introduced a T-square test. This test aims to preserve the intuitive appeal of Hopkins' (1954) test, and it proved more successful than Holgate's (1965) test. It consistsof the following statistic

$$h_{BG1} = \frac{1}{m} \sum \frac{2X^2}{2X^2+Z^2} \sim N(\frac{1}{2}, \frac{1}{12m})$$

$$h_{BG2} = \frac{2\sum X^2}{\sum Z^2} \sim F_{2m,2m} \qquad (A.9)$$

Both tests are comparatively powerful against regular or aggregated alternatives since relatively small or large values of h_{BG1} or h_{BG2} indicate a regular or aggregated pattern, respectively. Thus these tests can be used as three-way characterization of spatial point pattern as regular, random or aggregated corresponding to a small, non-significant or large value for h_{BG1} and h_{BG2}.

Diggle, et al. (1976) suggested CSR test with X and Z distance. Their statistic is as follows,

$$h_{DB} = \sum 2[\{\min(2X^2, Z^2)\}/(2X^2+Z^2)]/m \sim N(\frac{1}{2}, \frac{1}{12m}) \qquad (A.10)$$

It only detects nonrandomness and cannot distinguish between aggregation and regularity.

Cox & Lewis (1976) used the conditioned distance ratio method. They considered X and W distance and collected pairs $m_0 \leq m$ for which $W_i \leq X_i$. Transforming these two distances into r_i, they showed for regularity,

$$M = \min(r_1, \ldots, r_m) \sim Beta(1, m_0) \qquad (A.11)$$

and for aggregation,

$$R = \frac{\sum r}{m_0} \sim N(\frac{1}{2}, \frac{1}{12m_0}) \qquad (A.12)$$

Diggle (1977) suggested a generalized likelihood ratio test. This test is based on the method of Besag & Gleaves (1973) when he considered the difference between clustering and random heterogeneity in the environment.

$$h_D = 48m\{m\log[(\sum 2X^2+Z^2)/m] - \sum \log(2X^2+Z^2)\}/(13m+1)$$
$$\sim \chi^2_{m-1} \qquad (A.13)$$

It can be used as four-way characterization of spatial point patterns as regular, random-homogeneous, random-heterogeneous, or aggregated. This test can be used to test heterogeneity if the test of Besag & Gleaves (1973) doesn't have a satisfactory result.

Ripley (1977) considered an edge-corrected second-moment as a test based on the idea of Besag (1977). The second moment is

$$\hat{K}(t)=m^{-2}\sum k(x,y) \tag{A.14}$$

where $k(x,y)^{-1}$ is the proportion of the circumference of the circle centered x passing through y. Besag (1977) suggested the plot of the stabilizing function $L(t)=\sqrt{\hat{k}(t)/\pi}$ vs t and $L_m = sup|L(t)-t|$ as a test statistic for CSR.

Brown & Rothery (1978) considered three statistics,

$$D=\frac{\sum X^2}{m}$$

$$S=\frac{1}{m-1}\sum (X^2-D)^2/D^2$$

$$G=(\prod_{i=1}^{m} X^2)^{1/m}/D \tag{A.15}$$

here, D is the mean squared distance, S is the square of the coefficient of variation of the squared nearest neighbor distances. Small values indicate local regularity, for complete regularity S = 0. S is an equivalent form of the Eberhardt (1967) test except for the squared distances. G is the ratio of the geometric mean to the arithmetic mean of the squared distances. It lies in the interval (0, 1) and large values of G indicate local regularity. When points are chosen at random, the distribution of S and G depend on the number of points and on the shape, but not the size, of the area.

Hines & Hines (1979) modified the Eberhardt (1967) test and adapted it to T-square sampling,

$$h_{HH}=\frac{2m\sum(2X^2+Z^2)}{\{\sum[X\sqrt{2}+Z]\}^2} \tag{A.16}$$

They proved h_H is powerful when detecting aggregation.

Byth & Ripley (1980) used a semi-systematic sampling scheme as a modification of the Hopkins' (1954) test. Fig A-2 shows the schematic sampling of their procedure. Set up a grid of 2m points. From m sampling points, measure the squared distance to nearest event, $X_i^2, i=1,...,m$ with remaining points. Layout a small plot of a size which would contain about five events on average and count the events in each plot. Select m events at random from those enumerated and measure the squared distance W_i^2. Thus,

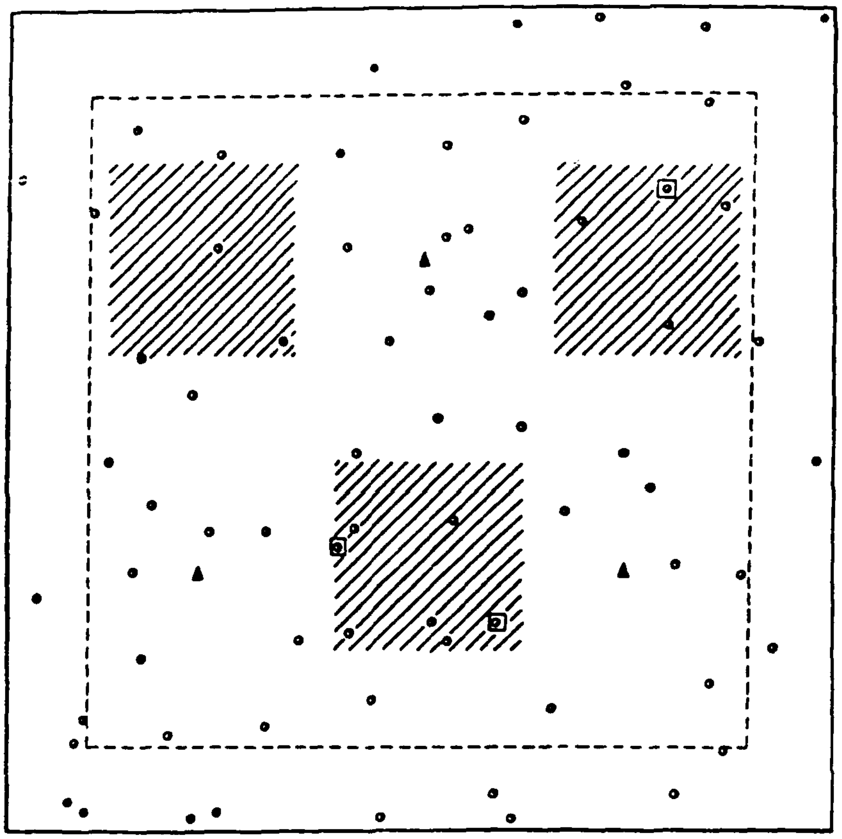


Figure A-2: Semisystematic Sampling Scheme of Byth & Ripley (1980)

$$h_{BR1}=\frac{\sum X^2}{\sum W^2} \sim F_{2m,2m}$$

$$h_{BR2}=\frac{1}{m}\sum\left(\frac{X^2}{X^2+W^2}\right) \sim N\left(\frac{1}{2},\frac{1}{12m}\right) \qquad (A.17)$$

and use these statistic as tests of CSR. They suggested h_{BR1} for suspecting clustered pattern and h_{BR2} for regular alternatives.

Heinrich (1984, 1986) modified the Ripley's (1977) second moment test and included the variance, assuming the normally distributed random variable of $\hat{K}(t)$. For testing the null hypothesis with given significance level α, he suggested

$$h_{HE}(t,\rho,a)=\frac{\rho a[\hat{K}(t)-\pi t^2]}{t\sqrt{2\pi(1+2\pi t^2\rho)}} \qquad (A.18)$$

where a is a length of a squared area.

A.2 Quadrat Count Tests

It is well known that the quadrat count test is a weak test compared with the distance-based test since (1) the size of the quadrat has effect on the results, (2) several different patterns may have same test results and (3) the counts lack spatial autocorrelations.

To test CSR with quadrat count method, we assume that the data are made up of independent counts $n_1,\ldots,n_m$ in m quadrats, each with area B, and with the mean of counts is $\bar{n}$

Fisher, et al. (1922) used the sample variance to mean ratio or index of dispersion as a test

$$I=\sum_{i=1}^{m}(n_i-\bar{n})^2/\{(m-1)\bar{n}\} \qquad (A.19)$$

Under CSR,

$$(m-1)I \sim \chi^2_{m-1} \qquad (A.20)$$

where, usually $m > 6$ and $\rho B > 1$. Significantly large or small values indicate aggregate or regular pattern, respectively.

Mead (19974) showed various alternative tests based on the hierarchical classification of quadrats. Upton & Fingleton (1985) described the use of quadrat count test in detail.

A.3 Discussion

We have discussed many tests which can be used to test CSR. To find the most efficient test set is not simple because each test has its own merit. In general, one needs to include edge effects. For this reason, we advocate the use of Ripley's (1977) test. In this type of test, any shape of region can be used and no correction for edge effects is required if we use the Monte-Carlo simulation suggested by Barnard (1963).

Since the CSR test can only be used to check the null hypothesis, we need an alternative model (i.e., non-Poisson point model) if the null hypothesis is not satisfied.

Appendix B
METHODS FOR EDGE-CORRECTION

When we calculate the second moment of given data points, we need to consider edge effects. Here, we discuss two popular methods which can be used to correct the edge effects.

B.1 Edge-correction with the Graphical Method

The original idea for this correction is Ripley's (1977) ; it was simplified by Ohser & Stoyan (1981) and by Diggle (1983). One definition of the second moment, K(t), is

$$\hat{K}(t)=n^{-2}A\sum k(x_1,x_2) \qquad (B.1)$$

where, n is the number of events in a region A and the sum is over all pairs (x_1,x_2) of events which are apart up to distance t. $k(x_1,x_2)$ is a correction factor for edge effects. Consider the circle centered on x_1 passing through x_2, see Fig. B-1. If this circle is completely within region A then k = 1. Otherwise, $1/k(x_1,x_2)$ is the proportion of the circumference of the circle within region.

Particularly, when a region A is rectangle with side a and b ($a < b$), Ohser & Stoyan (1981), considering the whole region, classified the distance measure as belonging to one of four cases, whereas, Diggle (1983), considering practical uses, grouped the distance measure in two cases. Ripley (1985) expanded these cases to irregular shaped regions and suggested a generalized graphical method.

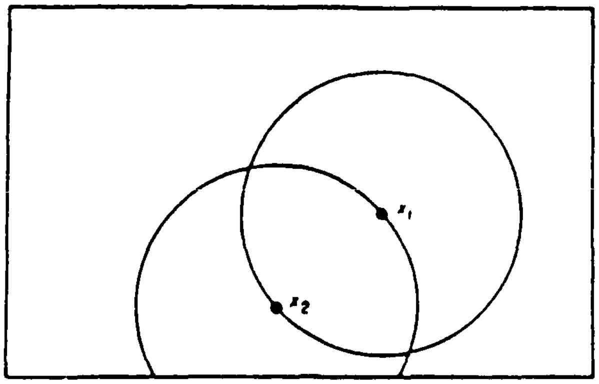


Figure B-1: Graphical Method for Edge-correction

B.2 Edge-correction by Toroidal Shift Method

Original toroidal shifting (Lotwick & Silverman, 1982) was used when evaluating an independence test of bivariate point processes. We can use this idea when we calculate the second moment. Assume that the region of interest is rectangle with sides a and b ($a < b$). Wrap the region onto a torus both in the x- and y-direction. Thus there are no edges. Calculate distance between two events as follows where we assume $x(x_1,x_2)$, $y(y_1,y_2)$

$$t(x,y)=\sqrt{[f_a(x_1-y_1)]^2+[f_b(x_2-y_2)]^2} \qquad (B.2)$$

where,

$$f_h(s)=\min(|s|,h-|s|) \qquad (B.3)$$

and calculate the second moment, K(t), with derived distance t. Here, K(t) is unbiased for all $t < t_0 = \min(a,b)/2$.

Appendix C
USER MANUAL AND LIST FOR "POINT" AND "FIBER"

C.1 Introduction

For the analysis of mapped fracture patterns, two different programs are needed :

- POINT : a program with which mid point behavior of mapped patterns and various point processes are analyzed.
- FIBER : a program which analyzes trace length behavior using a hierarchical concept.

In addition, a mathematical subroutine package called IMSL is used when generating random numbers and simulating basic distribution functions. Calculation of trace length distributions is done with program TDIST. Lastly, simulations of trace patterns using a termination probability are performed with program TERMN. These programs are installed both on the JVN super computer CYBER 205 and on the Micro-VAX II at MIT. All programs are based on FORTRAN 77.

C.2 Input Manual for Program POINT

Program POINT generates pseudo-random numbers, and from these, calculates inter-event and nearest-neighbor distances. It also computes second-moments of mapped data and simulation patterns. Free-format is used for all input data except for the title. Except for a default input data file (unit 5), an additional input data file is required when dealing with an inhomogeneous Poisson point process modelling and a doubly stochastic process modelling. In this case, the input file unit is 8.

1. TITLE (20A4)

2. IOPTN : analysis options

 1 : Inter-event distance analysis

 2 : Nearest-neighbor distance analysis

3 : Both 1 & 2

4 : Second-moment analysis

5 : Inhomogeneous Poisson point process model

6 : Marked point process model

7 : Doubly stochastic (Cox) point process model

3. If IOPTN = 7, then read ANGLE (in degrees). If not, skip, where ANGLE is a direction with which the angular second-moment is calculated.

4. NOPNT, NOSIM, NSTEP, IMAPP
   - NOPNT : No. of points generated
   - NOSIM : No. of simulations including analysis of mapped pattern
   - NSTEP : No. of calculation steps when evaluating the distance-based measurement
   - IMAPP = 0 : consider simulations only when checking validity of a model
   - IMAPP = 1 : consider simulations and mapped pattern when checking validity of a model

5. DSTEP : distance step length

6. KXCLS, KYCLS : No. of divisions in X- and Y-direction, respectively

7. XBOT, XTOP, YBOT, YTOP : boundary coordinates for X- and Y-direction

8. If IOPTN = 5, read values of curve-fitting parameter and options

XVAL1, XVAL2, YVAL1, YVAL2, CONST : parameters

JOPTN, IKERN

- JOPTN : analysis options

  1 : inter-event distance analysis

  2 : nearest-neighbor distance analysis

  3 : second-moment analysis

- IKERN

  1 : fixed kernel function measure

  2 : linear kernel function measure

SIGMA : standard deviation of a fixed kernel function (input unit = 8)

COORD(IOPNT,1) : X-coord. ; COORD(IOPNT,2) : Y-coord. (input unit = 8)

where, IOPNT = 1 through NOPNT

9. If IOPTN = 6, read IDUMM, COORD(IOPNT,1), COORD(IOPNT,2), TRACE(IOPNT)
    - IDUMM : dummy value
    - COORD(IOPNT,1) : X-coord.
    - COORD(IOPNT,2) : Y-coord.
    - TRACE(IOPNT) : trace length

10. If IOPTN = 7, read option NOJOB and coefficients
    - NOJOB

      NOJOB = 1 : Cox process with bivariate normal distribution function

      NOJOB = 2 : simulation of Cox process using angular second-moment analysis

      NOJOB = 3 : calculation of angular second-moment

    - XCOF2, YCOF2, FMULT, XVARI, YVARI, IPRIN (input unit = 8)

      XCOF2 : X-dir. spectral density coefficient

      YCOF2 : Y-dir. spectral density coefficient

      XVARI, YVARI : standard deviation for X- and Y-direction

      IPRIN = 0 : no printout ; = 1 print the results

    - DUMMY, DUMM1, IDUMM : dummy values
    - COORD(IOPNT,1) : X-coord.

      COORD(IOPNT,2) : Y-coord.

11. If IMAPP $\neq$ 0 and IOPTN $\leq$ 3, read coordinates of mapped pattern ; If not skip

COORD(IOPNT,1) : X-coord. ; COORD(IOPNT,2) : Y-coord.

IOPNT = 1 through NOPNT

C.3 Input Manual for Program FIBER

Program FIBER evaluates the hierarchical fiber (Line - segment) process which is a realization of the given mapped pattern.

1. TITLE (20A4)
2. NOPNT, NOFPT, NOLPT, NOSIM, NSTEP, DSTEP, SIGMA, TSTEP
    - NOPNT : No. of total fibers (or mid points)
    - NOFPT : No. of fibers in the 1st set
    - NOLPT : No. of fibers in the 2nd set
    - NOSIM : No. of simulations iterated
    - NSTEP : No. of steps when calculating the statistic
    - DSTEP : distance step size
    - SIGMA : standard deviation of the line-kernel function
    - TSTEP : unit of segment in a fiber
3. XBOT, XRANG, YBOT, YRANG
    - XBOT : X-coord. of starting boundary
    - XRANG : range of X-coord.
    - YBOT : Y-coord. of starting boundary
    - YRANG : range of Y-coord.
4. IOPTN, JOPTN, LOPTN

    IOPTN = 0 : do not enter the point process

    IOPTN = 1 : bivariate kernel function method

    IOPTN = 2 : independence test

    IOPTN = 3 : orientation correlation option

    IOPTN = 4 : MLE for orientation data

    JOPTN = 0 : do not enter the fiber process

    JOPTN = 1 : MLE for line-kernel function method

    JOPTN = 2 : MLE for nearest-neighbor fibers

    JOPTN = 3 : simulation of set 2 with line-kernel function method

    JOPTN = 4 : simulation of set 2 with nearest-neighbor distance method

LOPTN = 1 : calculate the bivariate second-moment

LOPTN = 2 : calculate the univariate second-moment

5. If IOPTN = 2 ; read coordinates of mid points of all fibers
   - COORD(IOPNT,1), COORD(IOPNT,2) : X- and Y-coord. of the mid point
   - ISHFT ; analysis option

     ISHFT = 1 : small perturbation test

     ISHFT = 2 : toroidal shift test
6. If IOPTN = 3 ; read ANGLE and coordinates of fibers
   - ANGLE : step angle magnitude in degree
   - COORD(IOPNT,1), COORD(IOPNT,2) : X- and Y-coord. at starting point of fiber

     COORD(IOPNT,3), COORD(IOPNT,4) : X- and Y-coord. at ending point of fiber
7. If IOPTN = 4 ; read coordinates of fibers of 2nd set and option MOPTN
   - COORD(IOPNT,1), COORD(IOPNT,2) : X- and Y-coord. at starting point of fiber

     COORD(IOPNT,3), COORD(IOPNT,4) : X- and Y-coord. at ending point of fiber
   - MOPTN : orientation distribution option

     MOPTN = 1 : Von Mises distribution on a circle

     MOPTN = 2 : wrapped normal distribution on a circle
8. If JOPTN ≤ 2, read XLAMO : distance measure at starting point
9. If JOPTN ≠ 0, read coordinate of 1st set of fibers
   - COORD(IOPNT,1), COORD(IOPNT,2) : X- and Y-coord. at starting point of fiber

     COORD(IOPNT,3), COORD(IOPNT,4) : X- and Y-coord. at ending point of fiber
10. If JOPTN = 1, = 2 or = 3 : read coord. of 2nd set of fibers
    - COORD(IOPNT,1), COORD(IOPNT,2) : X- and Y-coord. at starting point of fiber

      COORD(IOPNT,3), COORD(IOPNT,4) : X- and Y-coord. at ending point of fiber

C.4 Input Manual for Program TDIST

Prog:am TDIST estimates the trace length distribution using MLE. Currently, only the exponential form of the distribution is considered.

1. TRACE(IOPNT,1), TRACE(IOPNT,2), NCONF(IOPNT)
   - TRACE(IOPNT,1) : trace length for IOPNT-th trace
   - TRACE(IOPNT,2) : boundary length for IOPNT-th trace
   - NCONF(IOPNT) = 1 : trace length with both ends observable

     NCONF(IOPNT) = 2 : trace length with one end observable

     NCONF(IOPNT) = 3 : trace length with no end observable
2. NITER : No. of iterations (currently input as 1)
3. NOPTN : analysis option (currently input as 1)

C.5 Input Manual for Program TERMN

Program TERMN generates the trace pattern according to the simulated typical points. Also, it evaluates the termination points from calculated termination probabilities.

1. NOPNT, NOFPT, NOSIM, VMEAN, AMEAN, SIGMA, TRUNL
   - NOPNT : No. of mid points generated
   - NOFPT : No. of mid points of 1st set
   - NOSIM : No. of simulation performed
   - VMEAN : mean trace length of set 2
   - AMEAN : mean (strike) angle of set 2
   - SIGMA : standard deviation of angle of set 2
   - TRUNL : truncation length of set 2
2. XBOT, XTOP, YBOT, YTOP : coordinate of simulation boundary
3. MOPTN : orientation distribution option

   MOPTN = 1 : Von Mises distribution

   MOPTN = 2 : wrapped normal distribution
4. read coordinate of 1st set CORFP

CORFP(IOPNT,1), CORFP(IOPNT,2) : coordinate at starting point of fiber

CORFP(IOPNT,3), CORFP(IOPNT,4) : coordinate at ending point of fiber

5. IDUMM, COORD(IOPNT,1), COORD(IOPNT,2)
   - IDUMM : dummy value
   - COORD(IOPNT,1) : X-coord. of mid point of 2nd set
   - COORD(IOPNT,2) : Y-coord. of mid point of 2nd set

C.6 Program Listings : POINT, FIBER, TDIST, TERMN

```
      PROGRAM POINT
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C                                                             C
C     PROGRAM POINT GENERATES PSEUDO-RANDOM NUMBERS &         C
C     CALCULATE INTER-EVENT , NEAREST EVENT DISTANCES         C
C     AND SECOND-MOMENT OF THE                                C
C     SIMULATION DATA AS WELL AS MAPPED DATA                  C
C                                                             C
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *             NTOVL, IMAPP, IOPTN, XBOT,  XRANG, YBOT,
     *             YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
      DIMENSION KK(20000), AA(30000), TITLE(20)
C
C     COORD : COORDINATE OF MID-POINT (NOPNT,2)
C     RSCLE : NO. OF GENERATED RANDOM NUMBER (2*NOPNT)
C     NOFRE : NO. OF DISTANCE FREQUENCES IN A SIMULATION (NOSIM,NSTEP)
C     DISTS : DISTANCE MEASURE { 0.5*NOPNT*(NOPNT-1) }
C
      NRVAR = 30000
      NIVAR = 20000
C
      NDOFN = 2
      READ (5,800) TITLE
  800 FORMAT(20A4)
      WRITE(6,800) TITLE
C
C     1    ANALYSIS OPTION                              : IOPTN
C              1 : INTER-EVENT DISTANCES
C              2 : NEAREST NEIGHBOR DISTANCES
C              3 : BOTH 1 & 2
C              4 : SECOND MOMENT MEASURE DISTANCE
C              5 : INHOMOGENEOUS POISSON P.P.
C              6 : MARKED POINT PROCESS
C              7 : DOUBLY STOCHASTIC P.P.
C
```

```
C     1.1  ANGLE OF MEASUREMENT IN DEGREE                 : ANGLE
C           ( OPTIONAL WHEN IOPTN = 7 )
C
C     2    NO. OF MID-POINTS                              : NOPNT
C     3    NO. OF SIMULATIONS INCLUDING MAPPED DATA       : NOSIM
C     4    NO. OF DISTANCE STEPS                          : NSTEP
C     5    FLAG INDEX IMAPP                               : IMAPP
C               0 : SIMULATIONS ONLY
C               1 : INCLUDE A MAPPED PATTERN
C     6    DISTANCE STEP USED IN STATISTIC                : DSTEP
C
      NOANG = 1
      READ (5,*)    IOPTN
      WRITE(6,900) IOPTN
      IF ( IOPTN.EQ.7 ) THEN
              READ (5,*) ANGLE
              NOANG = 360 / ANGLE
              WRITE(6,980) ANGLE
      ENDIF
      READ (5,*)    NOPNT, NOSIM, NSTEP, IMAPP
      WRITE(6,910) NOPNT, NOSIM, NSTEP, IMAPP
      READ (5,*)    DSTEP
      WRITE(6,920) DSTEP
C
C     KXCLS = NO. DIVISIONS IN X-DIR.
C     KYCLS = NO. DIVISIONS IN Y-DIR.
C     USED IN INHOMOGENEOUS POISSON POINT PROCESS
C     IF IOPTN.NE.5, SET KXCLS = KYCLS = 1
C
      READ (5,*)    KXCLS,KYCLS
      WRITE(6,970) KXCLS,KYCLS
      NTOTL = NDOFN * NOPNT
      NTOVL = NOPNT * (NOPNT-1) / 2
      NTOST = NOSIM * NSTEP
      NTOSI = NOSIM * NOPNT
      KTCLS = KXCLS * KYCLS
      NSECT = NSTEP * NOANG
C
C     DYNAMIC DIMENSIONING
C
      NR1   = 1
      NR2   = NR1 + NTOTL
      NR3   = NR2 + NTOTL
      NR4   = NR3 + NTOVL
C
C     DFREQ = ( NOSIM,NOPNT )
C     SIMUL = ( KTCLS,2 )
C
      NR5   = NR4 + NTOSI
      NR6   = NR5 + KTCLS*2
C
C     SVALU = ( NOSIM,NSTEP ) : USED IN SECOND MOMENT MEASURE
C
      NR7   = NR6 + NTOST
      NR8   = NR7 + NOPNT
      NR9    = NR8 + NSECT
C
      NI1    = 1
      NI2    = NI1 + NTOST
C
      NRTOT = NR9 - 1
      NITOT = NI2 - 1
      IERRO = 0
C
      WRITE(6,930) NRTOT, NRVAR
      IF ( NRVAR.GT.NRTOT ) GO TO 20
      WRITE(6,940)
      IERRO = IERRO + 1
   20 CONTINUE
      WRITE(6,950) NITOT, NIVAR
      IF ( NIVAR.GT.NITOT ) GO TO 30
      WRITE(6,960)
      IERRO = IERRO + 1
   30 CONTINUE
      IF ( IERRO.GT.0 ) STOP
C
      DO 40 IVARI = 1,NRVAR
   40 AA(IVARI) = 0.
      DO 50 IVARI = 1,NIVAR
   50 KK(IVARI) = 0
C
C     IF BOTH INTER-EVENT AND NEAREST NEIGHBOR DISTANCE
C     METHOD ARE USED, OPEN TAPE10
C
      IF ( IOPTN.GE.3 ) THEN
           OPEN (10,FILE='TAPE10',FORM='UNFORMATTED',STATUS='NEW')
      ENDIF
C
C     CALL MAIN OPTION
C
      CALL MAINS (AA(NR1 ), AA(NR2 ), AA(NR3 ), AA(NR4 ), AA(NR5 ),
     *            AA(NR6 ), AA(NR7 ), AA(NR8 ), KK(NI1 ))
C
  900 FORMAT (//, 5X,'ANALYSIS OPTION                = ',I3,/,
     *            7X,'( 1 : INTER-EVENT DISTANCES  ) ',   /,
     *            7X,'( 2 : NEAREST EVENT DISTANCE ) ',   /,
     *            7X,'( 3 : BOTH 1 & 2 OPTIONS     ) ',   /,
     *            7X,'( 4 : 2nd MOMENT MEASURE     ) ',   /,
     *            7X,'( 5 : INHOMOGENEOUS POISSON P) ',   /,
     *            7X,'( 6 : MARKED POINT PROCESS   ) ',   /,
     *            7X,'( 7 : DOUBLY STOCHASTIC P.P. ) ',   /)
  910 FORMAT (//, 5X,'NO. OF MID-POINT GENERATED     = ',I3,/,
     *            5X,'NO. OF SIMULATIONS             = ',I3,/,
     *            5X,'NO. OF DISTANCE MEASURES       = ',I3,/,
     *            5X,'MAPPED PATTERN INDICATOR       = ',I3,/,
     *            7X,'( 0 : SIMPLE SIMULATIONS   )    ',    /,
     *            7X,'( 1 : MAPPED DATA INCLUDED )    ',    /,
     *        //)
  920 FORMAT ( /, 5X,'DISTANCE STEP SIZE             = ',F7.3,/)
  930 FORMAT ( /, 5X,'REAL STORAGE REQUIRED          = ',I7,
     *          /, 5X,'REAL STORAGE SPECIFIED        = ',I7  )
  940 FORMAT ( /, 3X,'***  INCREASE STORAGE FOR REAL ARRAYS ')
  950 FORMAT ( /, 5X,'INTEGER STORAGE REQUIRED      = ',I7,
     *          /, 5X,'INTEGER STORAGE SPECIFIED     = ',I7  )
  960 FORMAT ( /, 3X,'***  INCREASE STORAGE FOR INTEGER ARRAYS')
  970 FORMAT (//, 5X,'NO. OF SUBSET IN X-DIRECTION = ',I3,
     *          /, 5X,'NO. OF SUBSET IN Y-DIRECTION = ',I3,
     *         //)
  980 FORMAT (//, 5X,'ANGLE OF MEASUREMENT          = ',F5.1,' DEG.' )
C
      STOP
      END
CC
C
      SUBROUTINE MAINS ( COORD, RSCLE, DISTS, DFREQ, SIMUL, SVALU,
     *                   TRACF. ANMOM, NOFRE )
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *             NTOVL, IMAPP, IOPTN, XBOT,  XRANG, YBOT,
     *             YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
      DIMENSION COORD(NOPNT,2), RSCLE(NTOTL), DISTS(NTOVL),
     *          NOFRE(NOSIM,NSTEP), DFREQ(NOSIM,NOPNT),
     *          SIMUL(KTCLS,2), SVALU(NOSIM,NSTEP), TRACE(NOPNT),
     *          ANMOM(NOANG,NSTEP)
C
C     READ SIMULATION PATTERN SIZE
C
      READ (5,*) XBOT,XTOP,YBOT,YTOP
      XRANG = XTCP - XBOT
      YRANG = YTOP - YBOT
C
C     ANALYSIS OPTION
C
      GO TO (100,200,300,400,500,600,700) IOPTN
C
C     INTER-EVENT DISTANCE ANALYSIS
C
  100 CONTINUE
      CALL INTER ( COORD,RSCLE,DISTS,NOFRE )
      GO TO 800
C
C     NEAREST NEIGHBOR DISTANCE ANALYSIS
C
  200 CONTINUE
      CALL NEARS ( COORD,RSCLE,DISTS,NOFRE,DFREQ )
      GO TO 800
C
C     BOTH INTER-EVENT & NEAREST NEIGHBOR DISTANCES
C
  300 CONTINUE
      CALL INTER ( COORD,RSCLE,DISTS,NOFRE )
      CALL NEARS ( COORD,RSCLE,DISTS,NOFRE,DFREQ )
      GO TO 800
C
```

```
C     SECOND-MOMENT MEASURE METHOD
C
  400 CONTINUE
      CALL SMSTA ( COORD,RSCLE,SVALU )
      GO TO 800
C
C     INHOMOGENEOUS POISSON POINT PROCESS
C
  500 CONTINUE
      CALL IHPPS ( COORD,RSCLE,DISTS,SIMUL,NOFRE,DFREQ,SVALU )
      GO TO 800
C
C     MARKED POINT PROCESS
C     MARK : TRACE LENGTH OF EACH JOINT
C
  600 CONTINUE
      CALL MPPSS ( COORD,TRACE,SVALU )
      GO TO 800
C
C     DOUBLY STOCHASTIC POINT PROCESS
C
  700 CONTINUE
      CALL DOUBL ( COORD,RSCLE,DISTS,SIMUL,NOFRE,DFREQ,SVALU,ANMOM )
C
  800 CONTINUE
      RETURN
      END
CC
C
      SUBROUTINE INTER ( COORD,RSCLE,DISTS,NOFRE )
C
C     SUNROUTINE INTER CALCULATES THE INTER-EVENT DISTANCE
C     ANALYSIS
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *               NTOVL, IMAPP, IOPTN, XBOT,  XRANG, YBOT,
     *               YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
      DIMENSION COORD(NOPNT,2), RSCLE(NTOTL), DISTS(NTOVL),
     *          NOFRE(NOSIM,NSTEP)
C
      JOSIM = 1
C
      NPSIM = NOSIM
      IF ( IMAPP.GT.0 ) NPSIM = NOSIM - 1
      IOSIM = 1
      DSEED = SECNDS(0.0) * 100.
C
C     GENERATE MID-POINT COORDINATE USING RANDOM NUMBER
C     GENERATOR OF IMSL (GGUBS)
C
   10 CONTINUE
C
      CALL GGUBS ( DSEED, NTOTL, RSCLE )
      DO 20 IDOFN = 1,NOPNT
      JDOFN = IDOFN * 2 - 1
      KDOFN = IDOFN * 2
      COORD(IDOFN,1) = XBOTM + RSCLE(JDOFN) * XRANG
      COORD(IDOFN,2) = YBOTM + RSCLE(KDOFN) * YRANG
C
      IF ( IOSIM.EQ.1 ) THEN
             WRITE(6,930) (COORD(IDOFN,IODOF),IODOF=1,2)
      ENDIF
C
   20 CONTINUE
C
C     CALCULATE INTER-EVENT DITANCES
C
      CALL INDIS ( JOSIM,COORD,NOPNT,DSTEP,NSTEP,NOFRE,DISTS,
     *             NOSIM,NTOVL )
      IF ( IOSIM.LT.NPSIM ) THEN
           IOSIM = IOSIM + 1
           JOSIM = JOSIM + 1
           GO TO 10
      ENDIF
C
C     MAPPED PATTERN ANALYSIS
C
      IF ( IMAPP.GT.0 ) THEN
         DO 50 IOPNT = 1,NOPNT
         DO 50 IDOFN = 1,NDOFN
         COORD(IOPNT,IDOFN) = 0.
   50    CONTINUE
         DO 60 IOPNT = 1,NOPNT
         READ (5,*) ( COORD(IOPNT,IDOFN), IDOFN=1,NDOFN )
C
C        IF IOPTN = 3, SAVE COORD. DATA FOR LATER USE
C
         IF ( IOPTN.EQ.3 ) WRITE(10) COORD(IOPNT,1),COORD(IOPNT,2)
C
   60    CONTINUE
C
         IOSIM = NOSIM
         CALL INDIS ( IOSIM,COORD,NOPNT,DSTEP,NSTEP,NOFRE,
     *                DISTS,NOSIM,NTOVL )
      ENDIF
C
C     CALCULATE MIN. & MAX NO. OF DISTANCE MEASURE
C
      IF ( NOSIM.EQ.1 ) STOP
C
      MTOTL = 0
      WRITE(6,940)
      DO 40 ISTEP = 1, NSTEP
      MINNO = NOFRE(1,ISTEP)
      MAXNO = NOFRE(1,ISTEP)
      MTOTL = NOFRE(1,ISTEP)
      DO 30 IOSIM = 2,NPSIM
      IF ( NOFRE(IOSIM,ISTEP).LT.MINNO )
     *     MINNO = NOFRE(IOSIM,ISTEP)
```

```
      IF ( NOFRE(IOSIM,ISTEP) .GT.MAXNO )
     *     MAXNO = NOFRE(IOSIM,ISTEP)
      MTOTL = MTOTL + NOFRE(IOSIM,ISTEP)
   30 CONTINUE
C
      HMINV = MINNO
      HMAXV = MAXNO
      HAVRG = MTOTL
      HMINV = HMINV / NTOVL
      HMAXV = HMAXV / NTOVL
      HAVRG = HAVRG / ( NTOVL * NPSIM )
C
      DSTNS = DSTEP * ISTEP
C
C     EMPIRICAL SOLUTION
C
      HEXAT = 0.
      IF ( XRANG.EQ.YRANG ) THEN
        HEXAT = 3.1415 * DSTNS**2 - 8*DSTNS**3 /3 + DSTNS**4/2
      ELSE
        HEXAT = HAVRG
      ENDIF
C
C     MAPPED PATTERN CASE
C
      IF ( IMAPP.GT.0 ) THEN
           PAMAP = NOFRE(NOSIM,ISTEP)
           PAMAP = PAMAP / NTOVL
           WRITE(6,950) DSTNS,HMINV,HMAXV,HEXAT,PAMAP
           GO TO 40
      ENDIF
C
      WRITE(6,950) DSTNS,HMINV,HMAXV,HEXAT
   40 CONTINUE
C
  930 FORMAT (    5X,2(D15.7,2X))
  940 FORMAT (/, 10X,'INTER-EVENT DISTANCE METHOD             ',/,
     *            5X,'DISTANCE  MIN. VALUE   MAX. VALUE   EXACT',
     *               '     MAPPED',                            /)
  950 FORMAT (    5X,F7.3,5X,4(F7.5,3X) )
      RETURN
      END
CC
C
      SUBROUTINE INDIS ( IOSIM,COORD,NOPNT,DSTEP, NSTEP,NOFRE,
     *                   DISTS,NOSIM,NTOVL )
C
C     SUBROUTINE INDIS CALCULATES INTER-EVENT DISTANCES OF
C     MID-POINTS
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION COORD(NOPNT,2), NOFRE(NOSIM,NSTEP), DISTS(NTOVL)
C
      NPOIN = NOPNT - 1
      ICOUN = 0
      NCOUN = NOPNT * (NOPNT-1) / 2
      FSTEP = DSTEP
C
      DO 10 IPOIN = 1,NPOIN
      XCOOR = COORD(IPOIN,1)
      YCOOR = COORD(IPOIN,2)
      KPOIN = IPOIN + 1
      DO 10 JPOIN = KPOIN,NOPNT
      ICOUN = ICOUN + 1
      XCOR1 = COORD(JPOIN,1)
      YCOR1 = COORD(JPIN,2)
      DISTS(ICOUN) = SQRT((XCOOR-XCOR1)**2+(YCOOR-YCOR1)**2)
   10 CONTINUE
C
C     CALCULATES THE FREQUENCIES OF DISTANCE CATEGORIES
C
      KCOUN = 0
   20 CONTINUE
      JCOUN = 0
      KCOUN = KCOUN + 1
      FSTEP = DSTEP * KCOUN
      DO 30 ICOUN = 1,NCOUN
      IF ( DISTS(ICOUN).LE.FSTEP ) THEN
         JCOUN = JCOUN + 1
      ENDIF
   30 CONTINUE
      NOFRE(IOSIM,KCOUN) = JCOUN
      IF ( KCOUN.LT.NSTEP ) GO TO 20
      RETURN
      END
CC
C
      SUBROUTINE NEARS ( COORD,RSCLE,DISTS,NOFRE,DFREQ )
C
C     SUBROUTINE NEARS COMPUTE NEAREST NEIGHBOR DISTANCE
C     OF GIVEN EVENT
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *             NTOVL, IMAPP, IOPTN, XBOT,  XRANG, YBOT,
     *             YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
      DIMENSION COORD(NOPNT,2), RSCLE(NTOTL), DISTS(NTOVL),
     *          NOFRE(NOSIM,NSTEP), DFREQ(NOSIM,NOPNT)
C
C     INITIALIZE
C
      DO 110 ICOOR = 1,NOPNT
      DO 110 IDOFN = 1,NDOFN
      COORD(ICOOR,IDOFN) = 0.
  110 CONTINUE
      DO 120 ITOTL = 1,NTOTL
      RSCLE(ITOTL) = 0.
  120 CONTINUE
      DO 130 ITOVL = 1,NTOVL
      DISTS(ITOVL) = 0.
  130 CONTINUE
      DO 140 LOSIM = 1,NOSIM
      DO 140 LSTEP = 1,NSTEP
      NOFRE(LOSIM,LSTEP) = 0
  140 CONTINUE
C
      REWIND 10
C
      JOSIM = 1
C
      IF ( IMAPP.GT.0 ) NPSIM = NOSIM - 1
      IOSIM = 1
      DSEED = SECNDS(0.0) * 100.0
C
C     GENERATE MID-POINT COORDINATE USING RANDOM NUMBER
C     GENERATOR OF IMSL (GGUBS)
C
   10 CONTINUE
      CALL GGUBS ( DSEED, NTOTL, RSCLE )
      DO 20 IDOFN = 1,NOPNT
      JDOFN = IDOFN * 2 - 1
      KDOFN = IDOFN * 2
      COORD(IDOFN,1) = XBOT + RSCLE(JDOFN) * XRANG
      COORD(IDOFN,2) = YBOT + RSCLE(KDOFN) * YRANG
   20 CONTINUE
C
C     CALCULATE NEAREST NEIGHBOR DISTANCE
C
      CALL NEDIS ( IOSIM,COORD,NOPNT,DSTEP,NSTEP,NOFRE,DISTS,
     *             NOSIM,NTOVL,XRANG,YRANG,DFREQ,AVRGD )
      IF ( IOSIM.LT.NPSIM ) THEN
           IOSIM = IOSIM + 1
           GO TO 10
      ENDIF
C
C     MAPPED PATTERN ANALYSIS
C
      IF ( IMAPP.GT.0 ) THEN
        DO 30 IOPNT = 1,NOPNT
        DO 30 IDOFN = 1,NDOFN
        COORD(IOPNT,IDOFN) = 0.
   30   CONTINUE
        IF ( IOPTN.EQ.2 ) THEN
          DO 40 IOPNT = 1,NOPNT
          READ (5,*) ( COORD(IOPNT,IDOFN), IDOFN=1,NDOFN)
   40     CONTINUE
        ELSE
          WRITE(6,992)
          DO 45 IOPNT = 1,NOPNT
          READ (10) ( COORD(IOPNT,IDOFN), IDOFN=1,NDOFN)
   45   CONTINUE
        ENDIF
C
        IOSIM = NOSIM
        CALL NEDIS ( IOSIM,COORD,NOPNT,DSTEP,NSTEP,NOFRE,
     *               DISTS,NOSIM,NTOVL,XRANG,YRANG,DFREQ,
     *               AVRGD )
        AVRGD = AVRGD / ( NOPNT * ( NOPNT - 1 ) )
      ENDIF
C
C     CALCULATE MIN. & MAX NO. OF DISTANCE MEASURE
C
      IF ( NOSIM.EQ.1 ) STOP
C
      WRITE(6,940)
      DO 60 ISTEP = 1,NSTEP
      MINNO = NOFRE(1,ISTEP)
      MAXNO = NOFRE(1,ISTEP)
      DO 50 IOSIM = 2,NPSIM
      IF ( NOFRE(IOSIM,ISTEP).LT.MINNO )
     *     MINNO = NOFRE(IOSIM,ISTEP)
      IF ( NOFRE(IOSIM,ISTEP).GT.MAXNO )
     *     MAXNO = NOFRE(IOSIM,ISTEP)
   50 CONTINUE
C
      HMINV = MINNO
      HMAXV = MAXNO
      HMINV = HMINV / NOPNT
      HMAXV = HMAXV / NOPNT
C
      DSTNS = DSTEP * ISTEP
C
C     AVERAGE VALUE OF SIMULATED DATA
C
         PAVRG = 0.
         DO 70 IOSIM = 1,NPSIM
         PAVRG = PAVRG + NOFRE(IOSIM,ISTEP)
   70    CONTINUE
         PAVRG = PAVRG / ( NPSIM * NOPNT )
C
C        MAPPED PATTERN
C
      IF ( IMAPP.GT.0 ) THEN
         PAMAP = NOFRE(NOSIM,ISTEP)
         PAMAP = PAMAP / NOPNT
         WRITE(6,950)  DSTNS,HMINV,HMAXV,PAVRG,PAMAP
         GO TO 60
      ENDIF
C
      WRITE(6,950) DSTNS,HMINV,HMAXV,PAVRG
   60 CONTINUE
C
C     CALCULATE MONTE-CARLO STATISTIC
C
      IF ( IMAPP.GT.0 ) THEN
         RMINV = AMAX1(XRANG,YRANG)
         RMAXV = 0.
         WRITE(6.960)
         WRITE(6,965) AVRGD
         DO 100 IOSIM = 1,NOSIM
         CSTAT = 0.
         RSTAT = 0.
         DO 90 ISTEP = 1,NSTEP
         JSTAT = 0
         ZSTAT = NOFRE(IOSIM,ISTEP)
         DO 80 JOSIM = 1,NOSIM
         IF ( JOSIM.EQ.IOSIM ) GO TO 80
             JSTAT = JSTAT + NOFRE(JOSIM,ISTEP)
   80    CONTINUE
         XSTAT = JSTAT
         XSTAT = XSTAT / ( NOPNT - 1 )
         CSTAT = (( ZSTAT - XSTAT ) / NOPNT )**2
         RSTAT = RSTAT + CSTAT
   90    CONTINUE
         IF ( IOSIM.NE.NOSIM.AND.RSTAT.LE.RMINV ) RMINV = RSTAT
         IF ( IOSIM.NE.NOSIM.AND.RSTAT.GE.RMAXV ) RMAXV = RSTAT
         WRITE(6,970) IOSIM,RSTAT
  100    CONTINUE
         WRITE(6,980) RMINV,RMAXV,RSTAT
      ENDIF
C
  940 FORMAT (/, 10X,'NEAREST NEIGHBOR DISTANCE METHOD           ',/,
     *            5X,'DISTANCE   MIN. VALUE    MAX. VALUE    AVERAGE',
     *                '     MAPPED',                              /)
  950 FORMAT (    5X,F7.3,5X,4(F7.5,5X)   )
  960 FORMAT (//, 3X,'*** MONTE-CARLO STATISTIC',//)
  965 FORMAT (    5X,'SAMPLE MEAN',6X,F15.7,/)
  970 FORMAT (    5X,'STATISTIC ',I5,' = ',F15.7)
  980 FORMAT (//, 3X,'EXTREME VALUES OF SIMULATION',/,
     *            5X,'MIN. VALUE OF STATISTIC = ',F15.7,/,
     *            5X,'MAX. VALUE OF STATISTIC = ',F15.7,/,
     *            5X,'MAPPED VALUE            = ',F15.7   )
  992 FORMAT (//,' MAPPED  COORD IN NEARS ',/)
      RETURN
      END
CC
C
      SUBROUTINE NEDIS ( IOSIM,COORD,NOPNT,DSTEP,NSTEP,NOFRE,
     *                   DISTS,NOSIM,NTOVL,XRANG,YRANG,DFREQ,
     *                   AVRGD )
C
C     SUBROUTINE NEDIS MEASURES THE NEAREST OTHER EVENT
C     WITHIN A DISTANCE OF GIVEN EVENT
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION COORD(NOPNT,2), NOFRE(NOSIM,NSTEP), DISTS(NTOVL),
     *          DFREQ(NOSIM,NOPNT)
C
      NPOIN = NOPNT - 1
      NCOUN = NOPNT - 1
      FSTEP = DSTEP
C
      DISMN = AMAX1(XRANG,YRANG)
      DISMX = 0.
      AVRGD = 0.
```

```
C
C     DISTANCE CALCULATION
C
      DO 10 IPOIN = 1,NOPNT
      ICOUN = 1
      XCOOR = COORD(IPOIN,1)
      YCOOR = COORD(IPOIN,2)
      DO 20 JPOIN = 1,NOPNT
      IF ( JPOIN.EQ.IPOIN ) GO TO 20
      XCOR1 = COORD(JPOIN,1)
      YCOR1 = COORD(JPOIN,2)
      DISTS(ICOUN) = SQRT((XCOOR-XCOR1)**2+(YCOOR-YCOR1)**2)
C
C     CALCULATE AVERAGE NEAREST NEIGHBOR DISTANCE
C
      IF ( IOSIM.EQ.NOSIM ) THEN
          AVRGD = AVRGD + DISTS(ICOUN)
      ENDIF
C
      IF ( DISTS(ICOUN).LE.DISMN ) DISMN = DISTS(ICOUN)
      IF ( DISTS(ICOUN).GE.DISMX ) DISMX = DISTS(ICOUN)
      ICOUN = ICOUN + 1
   20 CONTINUE
      DFREQ(IOSIM,IPOIN) = DISMN
      DISMN = AMAX1(XRANG,YRANG)
   10 CONTINUE
C
C     CALCULATE THE FREQUENCIES OF DISTANCE CATEGORIES
C
      JCOUN = 0
   30 CONTINUE
      JCOUN = JCOUN + 1
      KCOUN = 0
      FSTEP = DSTEP * JCOUN
      DO 40 IPOIN = 1,NOPNT
      IF ( DFREQ(IOSIM,IPOIN).LE.FSTEP ) THEN
          KCOUN = KCOUN + 1
      ENDIF
   40 CONTINUE
      NOFRE(IOSIM,JCOUN) = KCOUN
      IF ( JCOUN.LT.NSTEP ) GO TO 30
      RETURN
      END
CC
C
      SUBROUTINE IHPPS (COORD,RSCLE,DISTS,SIMUL,NOFRE,DFREQ,SVALU )
C
C     SUBROUTINE IHPPS SIMULATES NON-HOMOGENEOUS POISSON POINT PROCESS
C     WHEN MAPPED POINT PATTERN DO NOT FOLLW HOMOGENEOUS POISSON
C     POINT PROCESS
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *              NTOVL, IMAPP, IOPTN, XBOT,  XRANG, YBOT,
     *              YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
```

```
      DIMENSION COORD(NOPNT,2),RSCLE(NTOTL),SIMUL(KTCLS,2),
     *          DISTS(NTOVL),NOFRE(NOSIM,NSTEP),DFREQ(NOSIM,NOPNT),
     *          SVALU(NOSIM,NSTEP)
C
      REWIND 10
C
      XCRIT = 0.5
C
C     READ CURVE-FITTING PARAMETERS
C
      READ (5,*) XVAL1,XVAL2,YVAL1,YVAL2,CONST
C
C     JOPTN = 1 : INTER-EVENT DISTANCE METHOD
C             2 : NEAREST NEIGHBOR DISTANCE METHOD
C             3 : SECOND-MOMENT MEASURE METHOD
C
C     IKERL = 1 : FIXED KERNEL FUNCTION MEASURE
C             2 : LINEAR KERNEL FUNCTION
C
      READ (5,*) JOPTN,IKERL
      WRITE(6,990) JOPTN,IKERL
C
C     JITRT IS USED IN FIXED KERNEL FUNCTION OPTION
C     JITRT = 1 : READ COORDINATES OF GIVEN MAP
C           > 1 : SKIP
C
      JITRT = 1
C
      IXREG = ( XRANG ) / KXCLS
      IYREG = ( YRANG ) / KYCLS
C
      NPSIM = NOSIM - 1
      IOSIM = 0
C
      ICOUN = 0
      JCOUN = 0
      DO 35 IYCLS = 1,KYCLS
      JCOUN = JCOUN + 1
      YRSUS = IYREG * IYCLS
      YPSUS = IYREG * ( IYCLS - 1 )
      VALUE = (XVAL1 + XVAL2*IXCLS)*IXCLS + (YVAL1+YVAL2*IYCLS)*IYCLS
      RAMBD = EXP ( VALUE + CONST ) * NOPNT
      ILAMB = RAMBD
      XLAMB = RAMBD - ILAMB
      IF ( XLAMB.GE.XCRIT) ILAMB = ILAMB + 1
      ICOUN = ICOUN + ILAMB
      KCOUN = ICOUN - ILAMB + 1
C
      SIMUL(JCOUN,1) = JCOUN
      SIMUL(JCOUN,2) = XLAMB
C
      NOSUS = ILAMB * NDOFN
C
      CALL RANDP (NOSUS,RSCLE,IXCLS,IYCLS,IOSIM)
C
```

```
      IITER = 1
      DO 20 JLAMB = KCOUN,ICOUN
      IDOFN = IITER * 2 - 1
      JDOFN = IDOFN + 1
      COORD(JLAMB,1) = XBOT + RSCLE(IDOFN) * XRANG
      COORD(JLAMB,2) = YPSUS + RSCLE(JDOFN) * IYREG
      IITER = IITER + 1
   20 CONTINUE
   35 CONTINUE
C
      IF ( ICOUN.EQ.NOPNT ) GO TO 120
      IREMG = NOPNT - ICOUN
      NTOJM = KYCLS - 1
      DO 50 K = 1,NTOJM
      SUMMA = SIMUL(K,2)
      IRESI = K
      KP = K + 1
      DO 60 L = KP,KTCLS
      IF ( SUMMA.GT.SIMUL(L,2) ) GO TO 60
      SUMMA = SIMUL(L,2)
      IRESI = L
   60 CONTINUE
      DIST1 = SIMUL(IRESI,1)
      DIST2 = SIMUL(IRESI,2)
      SIMUL(IRESI,1) = SIMUL(K,1)
      SIMUL(IRESI,2) = SIMUL(K,2)
      SIMUL(K.1) = DIST1
      SIMUL(K,2) = DIST2
   50 CONTINUE
      DO 70 I = 1,IREMG
      SIMUL(I,2) = SIMUL(I,2) + 100.
   70 CONTINUE
C
   10 CONTINUE
C
  120 CONTINUE
      IOSIM = IOSIM + 1
      ICOUN = 0
      MCOUN = 0
      DO 90 IYCLS = 1,KYCLS
      JCOUN = 0
      MCOUN = MCOUN + 1
      YPSUS = IYREG * ( IYCLS - 1 )
      DO 85 LCOUN = 1,KYCLS
      XCOUN = MCOUN
      IF ( SIMUL(LCOUN,1).EQ.XCOUN ) GO TO 86
   85 CONTINUE
   86 CONTINUE
      VALUE = (XVAL1+XVAL2*IXCLS)*IXCLS+(YVAL1+YVAL2*IYCLS)*IYCLS
      RAMBD = EXP ( VALUE + CONST ) * NOPNT
      ILAMB = RAMBD
      XLAMB = RAMBD - ILAMB
      IF ( XLAMB.GE.XCRIT ) ILAMB = ILAMB + 1
      IF ( SIMUL(LCOUN,2).GE.100. ) ILAMB = ILAMB + 1
C
```

```
C     MODIFY DENSITY ACCORDING TO KERNEL FUNCTION
C
      IF ( IKERL.EQ.1 ) THEN
C
           DENSE = FLOAT(ILAMB) / ( XRANG*IYREG )
      ELSE
           DENSE = FLOAT(ILAMB) / FLOAT(NOPNT)
      ENDIF
C
      ICOUN = ICOUN + ILAMB
      KCOUN = ICOUN - ILAMB + 1
C
C     NOSUS = ILAMB * NDOFN
C
      NOSUS = 2
   88 CONTINUE
      CALL RANDP ( NOSUS,RSCLE,IXCLS,IYCLS,IOSIM )
C
C     CALCULATE X-DIR. INTENSITY FUNCTION
C
      XVALU = RSCLE(1) * XRANG
C
      IF ( IKERL.EQ.1 ) THEN
           CALL KERNEL ( NOPNT,XVALU,VALUE,JITRT )
           JITRT = JITRT + 1
      ELSE
           CALL CURVE ( XVALU,VALUE )
      ENDIF
C
      PINTE = VALUE / DENSE
C
      IF ( RSCLE(1).LE.PINTE ) THEN
           COORD(KCOUN,1) =  XBOT + RSCLE(1) * XRANG
           COORD(KCOUN,2) = YPSUS + RSCLE(2) * IYREG
           KCOUN = KCOUN + 1
           JCOUN = JCOUN + 1
      ENDIF
      IF ( JCOUN.LT.ILAMB ) GO TO 88
C
   90 CONTINUE
C
      IF ( IOSIM.EQ.1) THEN
           KCONT = KCOUN - 1
           WRITE(6,910) KCONT
           DO 40 IOPNT = 1,NOPNT
           WRITE(6,920) ( COORD(IOPNT,IDOFN),IDOFN=1,NDOFN)
   40      CONTINUE
      ENDIF
C
      GO TO ( 210,220,230 ) JOPTN
  210    CALL INDIS ( IOSIM,COORD,NOPNT,DSTEP,NSTEP,NOFRE,
     *                DISTS,NOSIM,NTOVL )
         GO TO 240
  220    CALL NEDIS ( IOSIM,COORD,NOPNT,DSTEP,NSTEP,NOFRE,
     *                DISTS,NOSIM,NTOVL,XRANG,YRANG,DFREQ,
     *                 AVRGD )
         GO TO 240
  230    CALL SMSTT ( IOSIM,COORD,SVALU )
C
  240 CONTINUE
      IF ( IOSIM.LT.NPSIM ) GO TO 10
C
C     MAPPED PATTERN ANALYSIS
C
      REWIND 8
      READ (8,*) SIGMA
      DO 110 IOPNT = 1,NOPNT
      READ (8,*)  ( COORD(IOPNT,IDOFN),IDOFN=1,NDOFN)
  110 CONTINUE
      IOSIM = NOSIM
      GO TO ( 250, 260, 270 ) JOPTN
  250    CALL INDIS ( IOSIM,COORD,NOPNT,DSTEP,NSTEP,NOFRE,
     *                DISTS,NOSIM,NTOVL )
         WRITE(6,950)
         GO TO 280
  260    CALL NEDIS ( IOSIM,COORD,NOPNT,DSTEP,NSTEP,NOFRE,
     *                DISTS,NOSIM,NTOVL,XRANG,YRANG,DFREQ,
     *                AVRGD )
         AVRGD = AVRGD / ( NOPNT * ( NOPNT - 1 ) )
         WRITE(6,930)
         GO TO 280
  270    CALL SMSTT ( IOSIM,COORD,SVALU )
  280 CONTINUE
C
C     GO TO SUB. KSTAT IN SECOND-MOMENT MEASURE CASE
C
      IF ( JOPTN.EQ.3 ) THEN
          CALL KSTAT ( SVALU,NOPNT,NOSIM,NSTEP,DSTEP )
          GO TO 300
      ENDIF
C
      DO 150 ISTEP = 1,NSTEP
      MTOTL = 0
      MINNO = NOFRE(1,ISTEP)
      MAXNO = NOFRE(1,ISTEP)
      MTOTL = NOFRE(1,ISTEP)
      DO 130 IOSIM = 2,NPSIM
      IF ( NOFRE(IOSIM,ISTEP).LT.MINNO ) MINNO = NOFRE(IOSIM,ISTEP)
      IF ( NOFRE(IOSIM,ISTEP).GT.MAXNO ) MAXNO = NOFRE(IOSIM,ISTEP)
      MTOTL = MTOTL + NOFRE(IOSIM,ISTEP)
  130 CONTINUE
C
      HMINV = MINNO
      HMAXV = MAXNO
      IF ( JOPTN.EQ.1 ) THEN
          NSTST = NTOVL
      ELSE
          NSTST = NOPNT
      ENDIF
      HEXAT = 0.
      HAVRG = MTOTL
      HMINV = HMINV / NSTST
      HMAXV = HMAXV / NSTST
      PAVRG = HAVRG / ( NTOVL * NPSIM )
      HEXAT = HAVRG
C
      DSTNS = DSTEP * ISTEP
C
C     AVERAGE VALUE OF SIMULATED DATA
C
      PAVRG = 0.
      DO 140 IOSIM = 1,NPSIM
      PAVRG = PAVRG + NOFRE(IOSIM,ISTEP)
  140 CONTINUE
      PAVRG = PAVRG / ( NPSIM * NOPNT )
C
C     MAPPED PATTERN
C
      PAMAP = NOFRE(NOSIM,ISTEP)
      PAMAP = PAMAP / NSTST
      IF ( JOPTN.EQ.1) THEN
         WRITE(6,940) DSTNS,HMINV,HMAXV,HEXAT,PAMAP
      ELSE
         WRITE(6,940) DSTNS,HMINV,HMAXV,PAVRG,PAMAP
      ENDIF
  150 CONTINUE
  300 CONTINUE
C
C     MONTE-CARLO STATISTICS
C
      WRITE(6,960)
      DO 180 IOSIM = 1,NOSIM
      CSTAT = 0.
      RSTAT = 0.
      DO 170 ISTEP = 1,NSTEP
      JSTAT = 0
      IF ( JOPTN.EQ.3 ) THEN
           ZSTAT = SVALU(IOSIM,ISTEP)
      ELSE
           ZSTAT = NOFRE(IOSIM,ISTEP)
      ENDIF
C
      DO 160 JOSIM = 1,NOSIM
      IF ( JOSIM.EQ.IOSIM ) GO TO 160
      IF ( JOPTN.EQ.3 ) THEN
           XSTAT = XSTAT + SVALU(JOSIM,ISTEP)
      ELSE
           JSTAT = JSTAT + NOFRE(JOSIM,ISTEP)
      ENDIF
  160 CONTINUE
      IF ( JOPTN.NE.3 ) XSTAT = JSTAT
      XSTAT = XSTAT / ( NOPNT - 1 )
      CSTAT = (( ZSTAT - XSTAT ) / NOPNT )**2
      RSTAT = RSTAT + CSTAT
  170 CONTINUE
```

```
C
      IF ( IOSIM.EQ.1 ) THEN
           RMINV = RSTAT
           RMAXV = RSTAT
      ENDIF
C
      IF ( IOSIM.NE.NOSIM.AND.RSTAT.LE.RMINV ) RMINV = RSTAT
      IF ( IOSIM.NE.NOSIM.AND.RSTAT.GE.RMAXV ) RMAXV = RSTAT
      WRITE(6,970) IOSIM,RSTAT
  180 CONTINUE
      WRITE(6,980) RMINV,RMAXV,RSTAT
      RETURN
C
  910 FORMAT (//,5X,'INHOMOGENEOUS POISSON POINT PATTERN',//,
     *           5X,'TOTAL NO. OF MID-POINT GENERATED =',I5,//)
  920 FORMAT (7X,2(F15.7,3X))
  930 FORMAT (//,7X,'INHOMOGENEOUS POISSON POINT PROCESS', /,
     *         /,9X,'NEAREST NEIGHBOR DISTANCE STATISTIC',//,
     *           5X,'DISTANCE   MIN. VALUE    MAX. VALUE    AVERAGE',
     *          '    MAPPED',                                //)
  940 FORMAT (   5X,F7.3,5X,4(F7.5,5X) )
  950 FORMAT (//,7X,'INHOMOGENEOUS POISSON POINT PROCESS', /,
     *         /,9X,'INTER-EVENT DISTANCE STATISTIC',      //,
     *           5X,'DISTANCE   MIN. VALUE    MAX. VALUE    AVERAGE',
     *          '    MAPPED',                                //)
  960 FORMAT (//,7X,'*** MONTE - CARLO STATISTICS ***',// )
  970 FORMAT (   5X,'STATISTIC',I5,' = ',F15.3)
  980 FORMAT (//,3X,'EXTREME VALUES OF SIMULATION',    /,
     *           5X,'MIN. VALUE OF STATISTIC = ',F15.3,/,
     *           5X,'MAX. VALUE OF STATISTIC = ',F15.3,/,
     *           5X,'MAPPED VALUE            = ',F15.3   )
  990 FORMAT (//,5X,'ANALYSIS OPTION = ',                    I3,
     *        //,7X,'( 1 : INTER-EVENT DISTANCE METHOD      )',
     *         /,7X,'( 2 : NEAREST NEIGHBOR DISTANCE METHOD )',
     *         /,7X,'( 3 : SECOND MOMENT MEASURE METHOD     )',
     *        //,5X,'KERNEL FUNCTION OPTION = ',             I3,
     *        //,7X,'( 1 : FIXED KERNEL FUNCTION            )',
     *         /,7X,'( 2 : LINEAR KERNEL FUNCTION           )')
      END
CC
C
      SUBROUTINE RANDP (NOSUS,RSCLE,IXCLS,IYCLS,IOSIM)
C
C     SUBROUTIN RANDP GENERATE PSUDO-RANDOM NUMBER
C     USING IMSL LIBRARY
C
      DIMENSION RSCLE(NOSUS)
C
      IF ( IYCLS.EQ.1.AND.IOSIM.EQ.0 )
     *       DSEED = SECNDS(0.0) * 100.0
C
      CALL GGUBS ( DSEED,NOSUS,RSCLE )
C
      RETURN
      END
```

```
CC
C
      SUBROUTINE CURVE ( XVALU,VALUE )
C
C     SUBROUTINE CURVE CALCULATE CURVE FITTING COEFFICIENTS
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      FUNC1 = EXP( -0.5*( (XVALU-3.  )/1.2  )**2 )
      FUNC1 = 0.368*FUNC1/( 1.2*2.5 )
      FUNC2 = EXP( -0.5*( (XVALU-9.2 )/2.03)**2 )
      FUNC2 = 0.86 *FUNC2/( 2.03*2.5 )
      FUNC3 = EXP( -0.5*( (XVALU-13.4 )/0.86)**2 )
      FUNC3 = 0.14*FUNC3 /(0.86 * 2.5 )
      FUNC4 = EXP( -0.5*( (XVALU-22.65)/0.93 )**2 )
      FUNC4 = 0.42*FUNC4/(0.93 * 2.5 )
      FUNC5 = EXP( -0.5*( (XVALU-34.35 )/4.3 )**2 )
      FUNC5 = 1.[illegible]2*FUNC5/(4.3*2.5)
      FUNC6 = EXP( -0.5*( (XVALU-44.7 )/4.4 )**2 )
      FUNC6 = 0.41*FUNC6/(4.4*2.5)
      VALUE = FUNC1 + FUNC2 + FUNC3 + FUNC4 + FUNC5 + FUNC6
      RETURN
      END
CC
C
      SUBROUTINE KERNEL ( NOPNT,XVALU,VALUE,JITRT )
C
C     SUBROUTINE KERNEL EVALUATE FIXED KERNEL FUNCTION OF
C     ANY GIVEN DATA POINT IN X-DIRECTION
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION COORD(106,2)
C
      IF ( JITRT.EQ.1 ) THEN
C
C     READ STANDARD DEVIATION OF THE KERNEL FUNCTION
C
          READ(8,*) SIGMA
C
C     READ COORDINATES OF GIVEN MAP
C

          DO 10 IOPNT = 1,NOPNT
          READ (8,*) (COORD(IOPNT,IDOFN),IDOFN=1,2)
   10     CONTINUE
C
          WRITE(6,920) SIGMA
      ENDIF
C
      PITWO = 6.283185
      DO 20 IOPNT = 1,NOPNT
      XMEAN = COORD(IOPNT,1)
      FUNCT = EXP ( -0.5* ( (XVALU - XMEAN)/SIGMA)**2 )
      VALUE = VALUE + FUNCT / ( SQRT(PITWO)*SIGMA )
C
```

```
C     CONSIDER EDGE EFFECTS
C
      IF ( XMEAN.LE.20..OR.XMEAN.GE.30. ) THEN
           FUNCT = EXP ( -0.5*( (XVALU + XMEAN)/SIGMA)**2 )
           VALUE = VALUE + FUNCT / ( SQRT(PITWO)*SIGMA )
      ENDIF
   20 CONTINUE
      VALUE = VALUE / NOPNT
C
  920 FORMAT(//,5X,'STD. DEV. IN KERNEL FUNCTION = ',F5.3,//)
      RETURN
      END
CC
C
      SUBROUTINE SMSTT ( IOSIM, COORD, SVALU, ANMOM, NOJOB )
C
C     SUBROUTINE SMSTT EVALUATE SECOND-MOMENT STATISTIC OF
C     MAPPED DATA
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *                NTOVL, IMAPP, IOPTN, XBOT,  XRANG, YBOT,
     *                YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
      DIMENSION COORD(NOPNT,2), SVALU(NOSIM,NSTEP),
     *             ANMOM(NOANG,NSTEP)
C
      NPSIM = NOSIM - 1
      NPPNT = NOPNT - 1
      CNSTV = 1.
C
      DO 30 ISTEP = 1,NSTEP
      SVALU(IOSIM,ISTEP) = 0.
      DO 30 IOANG = 1,NOANG
      ANMOM(IOANG,ISTEP) = 0.
   30 CONTINUE
C
C     CALCULATE WEIGHTING COEFFICIENT w(X,u)
C
      DO 20 IPOIN = 1,NOPNT
      XCOOR = COORD(IPOIN,1)
      YCOOR = COORD(IPOIN,2)
      DIST1 = AMIN1 ( XCOOR, (XRANG - XCOOR) )
      DIST2 = AMIN1 ( YCOOR, (YRANG - YCOOR) )
      SQDIS = DIST1**2 + DIST2**2
      DO 10 JPOIN = 1,NOPNT
      IF ( JPOIN.EQ.IPOIN ) GO TO 10
      XCOR1 = COORD(JPOIN,1)
      YCOR1 = COORD(JPOIN,2)
      DISTO = SQRT((XCOOR-XCOR1)**2+(YCOOR-YCOR1)**2)
      WEIGT = 0.
      ISTEP = DIST0 / DSTEP + 1
      IF ( ISTEP.GT.NSTEP ) GO TO 10
      IF ( DIST0**2.LE.SQDIS ) THEN
           DIST3 = ACOS( CNSTV * AMIN1(DIST1,DIST0) / DIST0 )
           DIST4 = ACOS( CNSTV * AMIN1(DIST2,DIST0) / DIST0 )
           WEIGT = 1. - ( DIST3 + DIST4 ) / 3.14159
      ELSE
           DIST3 = ACOS( CNSTV * DIST1 / DIST0 )
           DIST4 = ACOS( CNSTV * DIST2 / DIST0 )
           WEIGT = 0.75 - ( DIST3 + DIST4 ) / 6.28318
      ENDIF
      WEIGT = XRANG * YRANG / ( WEIGT * (NOPNT)**2 )
C
C     CALCULATE K FUNCTION
C
      SVALU(IOSIM,ISTEP) = SVALU(IOSIM,ISTEP) + WEIGT
C
C     CALCULATE ANGULAR K-FUNCTION
C
      IF ( NOJOB.NE.3 ) GO TO 10
      IF ( IOSIM.EQ.NOSIM ) THEN
           ANGL2 = ATAN( (YCOR1-YCOOR)/(XCOR1-XCOOR) )
           IF( ANGL2.LT.0. ) THEN
               ANGL1 = 360. + ANGL2
               IF ( XCOR1.LT.XCOOR ) ANGL1 = 180. + ANGL2
           ELSE
               ANGL1 = ANGL2
               IF ( XCOR1.LT.XCOOR ) ANGL1 = 180. + ANGL2
           ENDIF
           IOANG = ANGL1 / ANGLE + 1
           WEIGT = XRANG * YRANG / ( NOPNT**2 )
           ANMOM(IOANG,ISTEP) = ANMOM(IOANG,ISTEP) + WEIGT
      ENDIF
C
   10 CONTINUE
   20 CONTINUE
C
C     CUMULATIVE K FUNCTION
C
   90 CONTINUE
      DO 40 ISTEP = 1,NSTEP
      ADDTV = 0.
      IF ( ISTEP.EQ.1 ) GO TO 40
      KSTEP = ISTEP - 1
C
      ADDTV = SVALU(IOSIM,KSTEP)
      SVALU(IOSIM,ISTEP) = SVALU(IOSIM,ISTEP) + ADDTV
   40 CONTINUE
C
      IF ( IOSIM.NE.NOSIM ) RETURN
C
C     CUMULATIVE ANGULAR K-FUNCTION
C
      IF ( NOJOB.NE.3 ) RETURN
      DO 70 JOANG = 1, NOANG
      DO 60 ISTEP = 1, NSTEP
      ADDTV = 0.
      IF ( ISTEP.EQ.1) GO TO 60
      KSTEP = ISTEP - 1
      ADDTV = ANMOM(JOANG,KSTEP)
      ANMOM(JOANG,ISTEP) = ANMOM(JOANG,ISTEP) + ADDTV
   60 CONTINUE
   70 CONTINUE
      RETURN
      END
CC
C
      SUBROUTINE KSTAT(SVALU,NOPNT,NOSIM,NSTEP,DSTEP,XRANG,YRANG,ANMOM,
     *                 ANGLE,NOANG,NOJOB )
C
C     SUBROUTINE KSTST INTERPRET THE K FUNCTION
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      DIMENSION SVALU(NOSIM,NSTEP), ANMOM(NOANG,NSTEP), ANCOV(100)
C
C     FIND MIN. & MAX. VALUE OF SIMULATED DATA
C
      AREAT = XRANG * YRANG
      XLAMB = NOPNT / AREAT
      XLAM2 = XLAMB**2
C
      WRITE(6,910)
      NPSIM = NOSIM - 1
      PHI'U = 3.141592
C
      DO 20 ISTEP = 1,NSTEP
      STOTL = 0.
      IF (NOJOB.EQ.3) GO TO 15
      SMINV = SVALU(1,ISTEP)
      SMAXV = SVALU(1,ISTEP)
      STOTL = SVALU(1,ISTEP)
      DO 10 IOSIM = 2,NPSIM
      IF ( SVALU(IOSIM,ISTEP).LT.SMINV ) SMINV = SVALU(IOSIM,ISTEP)
      IF ( SVALU(IOSIM,ISTEP).GT.SMAXV ) SMAXV = SVALU(IOSIM,ISTEP)
      STOTL = STOTL + SVALU(IOSIM,ISTEP)
   10 CONTINUE
      SAVRG = STOTL / NPSIM
C
C     MAPPED PATTERN
C
   15 CONTINUE
C
      SMAPP = SVALU(NOSIM,ISTEP)
C
C     STABILIZE THE K VALUE
C
      DSTNS = DSTEP * ISTEP
      DSTS2 = DSTNS **2
      SMINV = SMINV - PHIVU*DSTS2
      SMAXV = SMAXV - PHIVU*DSTS2
      SAVRG = SAVRG - PHIVU*DSTS2
      SMAPP = SMAPP - PHIVU*DSTS2
C
```

```
      IF (NOJOB.EQ.3) THEN
        WRITE(6,920)DSTNS,SVALU(NOSIM,ISTEP)
      ELSE
        WRITE(6,920)DSTNS,SMINV,SMAXV,SAVRG,SMAPP
      ENDIF
   20 CONTINUE
      IF ( NOJOB.NE.3 ) RETURN
C
C     CALCULATE ANISOTROPIC COVARIANCE FUNCTION
C
      WRITE(6,930)
      NOAN1 = NOANG / 2
      NSTP1 = NSTEP - 1
C
      DO 25 ISTEP = 1,NSTEP
      DO 26 IOANG = 1,NOAN1
      JOANG = IOANG + NOAN1
      ANMOM(IOANG,ISTEP) = ANMOM(IOANG,ISTEP) + ANMOM(JOANG,ISTEP)
   26 CONTINUE
      DO 27 KANGL = 1,18
      ANMOM(KANGL,ISTEP) = ANMOM(KANGL,ISTEP) * 18.
   27 CONTINUE
      DSTS1 = DSTEP * ISTEP
      SVUTN = SVALU(NOSIM,ISTEP)
      THERY = PHIVU * DSTS1**2
   25 CONTINUE
C
      WRITE(6,950)
      WRITE(6,970)
      XLAMB = NOPNT / AREAT
      ANGCT = 1.
      DO 70 ISTEP = 2,NSTEP
      JSTEP = ISTEP - 1
      DSTNS = JSTEP * DSTEP
C
      ANCO1 = ANMOM(4,ISTEP) - ANMOM(4,JSTEP)
      ANCO2 = ANMOM(13,ISTEP)- ANMOM(13,JSTEP)
      DKFUN = SVALU(NOSIM,ISTEP) - SVALU(NOSIM,JSTEP)
      COEF2 = PHIVU * DSTNS * 2. * DSTEP
C
      ANCO1 = ANCO1 / COEF2
      ANCO2 = ANCO2 / COEF2
      DKFUN = DKFUN / COEF2
C
      COVA1 = ANCO1 - 1.
      COVA2 = ANCO2 - 1.
      COVA3 = DKFUN - 1.
      WRITE(6,980) DSTNS,ANCO1,ANCO2,DKFUN,COVA1,COVA2,COVA3
   70 CONTINUE
C
  910 FORMAT(//,7X,'INHOMOGENEOUS POISSON POINT PROCESS', /,
     *        /,7X,'SECOND-MOMENT MEASUREMENT STATISTIC',//,
     *          3X,'DISTANCE  MIN. VALUE  MAX. VALUE',
     *          2X,'AVERAGE  MAPPED    K-ft   COVARIANCE',            //)
  920 FORMAT(   1X,F9.3,1X,5(F9.3,1X),F15.6 )
  930 FORMAT(//,7X,'ANGULAR K - FUNCTION',/,5X,'DIST.',
     *           2X,'K(40)     K(130)     K(MAPPED)     K(ISO.)'/)
  950 FORMAT(//,7X,'ANGULAR K-MOMENT MEASURE',/)
  970 FORMAT(//,5X,'COVARIANCE FUNCTION',/,
     *        7X,' DIST.     COV(r,40.)    COV(r,130)    COV(r,360.)',/)
  980 FORMAT(2X,F6.3,6(F9.5,1X))
C
      RETURN
      END
CC
C
      SUBROUTINE SMSTA ( COORD,RSCLE,SVALU )
C
C     SUBROUTINE SMSTA CALCULATE SECOND MOMENT MEASURE DISTANCE
C     OF MID-POINT MAP
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *             NTOVL, IMAPP, IOPTN, XBOT,  XRANG, YBOT,
     *             YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
      DIMENSION COORD(NOPNT,2), RSCLE(NTOTL), SVALU(NOSIM,NSTEP)
C
      NPSIM = NOSIM
      IF ( IMAPP.GT.0 ) NPSIM = NOSIM - 1
      IOSIM = 1
      DSEED = SECNDS(0.0) * 100.
C
C     GENERATE MID-POINT COORDINATE USING RANDOM NUMBER
C     GENERATOR OF IMSL ( GGUBS )
C
      WRITE(6,900)
   10 CONTINUE
      CALL GGUBS ( DSEED, NTOTL,RSCLE )
      DO 20 IDOFN = 1,NOPNT
      JDOFN = IDOFN * 2 - 1
      KDOFN = IDOFN * 2
      COORD(IDOFN,1) = XBOT + RSCLE(JDOFN) * XRANG
      COORD(IDOFN,2) = YBOT + RSCLE(KDOFN) * YRANG
C
      IF ( IOSIM.EQ.1 ) THEN
           WRITE(6,910) (COORD(IDOFN,IODOF),IODOF=1,2)
      ENDIF
C
   20 CONTINUE
C
C     CALCULATE SECOND-MOMENT MEASURE
C
      CALL SMSTT ( IOSIM, COORD, SVALU )
C
      IF ( IOSIM.LT.NPSIM ) THEN
           IOSIM = IOSIM + 1
           GO TO 10
      ENDIF
C
C     MAPPED PATTERN ANALYSIS
```

```
C
      IF ( IMAPP.GT.0 ) THEN
           DO 50 IOPNT = 1,NOPNT
           DO 50 IDOFN = 1,NDOFN
           COORD(IOPNT,IDOFN) = 0.
   50      CONTINUE
           DO 60 IOPNT = 1,NOPNT
           READ (5,*) ( COORD(IOPNT,IDOFN),IDOFN=1,NDOFN )
   60      CONTINUE
C
           IOSIM = NOSIM
           CALL SMSTT ( IOSIM, COORD, SVALU )
      ENDIF
C
C     CALCULATE MIN.  & MAX. NO. OF DISTANCE MEASURE
C
      IF ( NOSIM.EQ.1 ) RETURN
      CALL KSTAT ( SVALU,NOPNT,NOSIM,NSTEP,DSTEP,XRANG,YRANG)
C
  900 FORMAT (//,5X,'SECOND MOMENT MEASURE DISTANCE METHOD',//)
  910 FORMAT ( 2(F15.7,5X) )
C
      RETURN
      END
CC
C
      SUBROUTINE MPPSS ( COORD,TRACE,SVALU )
C
C     SUBROUTINE MPPSS EVALUATE MARKED POINT PROCESS AND
C     CALCULATE K-FUNCTION OF MARKS
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *             NTOVL, IMAPP, IOPTN, XBOT,  XRANG, YBOT,
     *             YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
      DIMENSION COORD(NOPNT,2),TRACE(NOPNT),
     *          SVALU(NOSIM,NSTEP)
C
C     READ MID-POINT COORDINATE & TRACE LENGTH
C
      TMEAN = 0.
      DO 5 IOPNT = 1,NOPNT
      READ (7,*) IDUMM,(COORD(IOPNT,IDOFN),IDOFN=1,NDOFN),TRACE(IOPNT)
      TMEAN = TMEAN + TRACE(IOPNT)
    5 CONTINUE
      TMEAN = TMEAN / NOPNT
      WRITE(6,999) TMEAN
      TMEN2 = TMEAN**2
C
C     INITIALIZE K-FUNCTION & M.P.P. K-FUNCTION
C
      DO 30 ISTEP = 1,NSTEP
      SVALU(1,ISTEP) = 0.
      SVALU(2,ISTEP) = 0.
   30 CONTINUE
```

```
C
C     CONSIDER EDGE EFFECT
C
      DO 20 IPOIN = 1,NOPNT
      XCOOR = COORD(IPOIN,1)
      YCOOR = COORD(IPOIN,2)
      DIST1 = AMIN1 ( XCOOR,  (XRANG - XCOOR) )
      DIST2 = AMIN1 ( YCOOR,  (YRANG - YCOOR) )
      SQDIS = DIST1**2 + DIST2**2
      DO 10 JPOIN = 1,NOPNT
      IF ( JPOIN.EQ.IPOIN ) GO TO 10
      XCOR1 = COORD(JPOIN,1)
      YCOR1 = COORD(JPOIN,2)
      DIST0 = SQRT((XCOOR-XCOR1)**2+(YCOOR-YCOR1)**2)
      WEIGT = 0.
      ISTEP = DIST0 / DSTEP + 1
      IF ( ISTEP.GT.NSTEP ) GO TO 10
      IF ( DIST0**2.LE.SQDIS ) THEN
           DIST3 = ACOS( AMIN1(DIST1,DIST0) / DIST0 )
           DIST4 = ACOS( AMIN1(DIST2,DIST0) / DIST0 )
           WEIGT = 1. - ( DIST3 + DIST4 ) / 3.14159
      ELSE
           DIST3 = ACOS( DIST1 / DIST0 )
           DIST4 = ACOS( DIST2 / DIST0 )
           WEIGT = 0.75 - ( DIST3 + DIST4 ) / 6.28318
      ENDIF
C
      SVALU(1,ISTEP) = SVALU(1,ISTEP) + 1. / WEIGT
      SVALU(2,ISTEP) = SVALU(2,ISTEP) + 1. / WEIGT *
     *                 TRACE(IPOIN) * TRACE(JPOIN)
   10 CONTINUE
   20 CONTINUE
C
C     CUMULATIVE K & MARKED K FUNCTION
C
      WRITE(6,900)
      DO 40 ISTEP = 1,NSTEP
      SVALU(2,ISTEP) = SVALU(2,ISTEP) / SVALU(1,ISTEP)
      DSTNS = DSTEP * ISTEP
      PAIRF = SVALU(2,ISTEP) / TMEN2
      PAIRC = SVALU(2,ISTEP) / SVALU(1,ISTEP)
      WRITE(6,910) DSTNS,PAIRC,PAIRF
   40 CONTINUE
C
  900 FORMAT(//,7X,'MARKED POINT PROCESS',/,
     *        /,5X,'  DISTANCE    CORRELATION F    STABILIZED F ')
  910 FORMAT(5X,3(F10.3,3X) )
  999 FORMAT(//,5X,'TMEAN = ',F15.7)
      RETURN
      END

CC
C
      SUBROUTINE DOUBL ( COORD,RSCLE,DISTS,SIMUL,NOFRE,DFREQ,SVALU,
     *                   ANMOM )
```

```
C
C     SUBROUTINE DOUBL ESTIMATES THE DOUBLY STOCHASTIC POINT
C     PROCESS.
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *             NTOVL, IMAPP, IOPTN, XBOT,  XRANG, YBOT,
     *             YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
      DIMENSION COORD(NOPNT,2), RSCLE(NTOTL), SIMUL(KTCLS,2),
     *          DISTS(NTOVL), NOFRE(NOSIM,NSTEP),
     *          DFREQ(NOSIM,NOPNT), SVALU(NOSIM,NSTEP),
     *          ANMOM(NOANG,NSTEP)
      DIMENSION RSCL1(5000),HCOOR(5000,2)
C
      REWIND 10
      IFLAG = 0
      JITRT = 1
      NPSIM = NOSIM - 1
C
C     ANALYSIS OPTION
C       NOJOB = 1 : DO COX PROCESS WITH BIVARIATE NORMAL KERNEL FUNCTION
C       NOJOB = 2 : USING ANGULAR K-FUNCTION SIMULATE THE COX PROCESS
C       NOJOB = 3 : CALCULATE THE ANGULAR K-FUNCTION
C
      READ(5,*) NOJOB
      GO TO ( 5, 100, 35, 5 ), NOJOB
C
C     ITERATE FOR EACH SIMULATIONS
C
    5 CONTINUE
      DO 30 IOSIM = 1,NPSIM
      IF (NOJOB.EQ.4) GO TO 6
      IF ( IOSIM.EQ.1 ) THEN
C
C     CALCULATE THE MAX. DENSITY COEFFICIENT IN AN AXI-SYMMETRIC
C     NORMAL DENSITY FUNCTION.
C
         CALL MAXCF ( COEMX, NOPNT, XRANG, YRANG, XBOT, YBOT )
C
C     GENERATE THE HOMOGENEOUS POISSON POINT PATTERN ACCORDING TO
C     THE MAX. DENSITY FUNCTION.
C     NOGPT : NO. OF  POINTS GENERATED
C
         AREAT = XRANG * YRANG
         NOGPT = COEMX * AREAT
         NOGP2 = 2 * NOGPT
         DSEED = 123457.D0
      ENDIF
C
   10 CONTINUE
      CALL GGUBS ( DSEED, NOGP2, RSCL1 )
      DO 20 IDOFN = 1, NOGPT
      JDOFN = IDOFN + NOGPT
      HCOOR(IDOFN,1) = XBOT + RSCL1(IDOFN) * XRANG
      HCOOR(IDOFN,2) = YBOT + RSCL1(JDOFN)  * YRANG
   20 CONTINUE
C
C     DO THINNING PROCESS AND GET A COX PATTERN
C
      CALL THINP ( NOGPT, HCOOR, COORD, NOPNT, COEMX, IOSIM,
     *             XRANG, YRANG, XBOT,  YBOT  )
C
C     2nd MOMENT MEASURE
C
    6 CALL PAREN ( COORD, IOSIM )
      CALL SMSTT ( IOSIM, COORD, SVALU, ANMOM, NOJOB )
   30 CONTINUE
      GO TO 85
C
C     MAPPED PATTERN ANALYSIS
C
   35 CONTINUE
      REWIND 8
      READ (8,*) DUMMY, DUMM1, IDUMM
      DO 40 IOPNT = 1, NOPNT
      READ (8,*)  ( COORD(IOPNT,IDOFN),IDOFN=1,NDOFN )
   40 CONTINUE
      IOSIM = NOSIM
      CALL SMSTT ( IOSIM, COORD, SVALU, ANMOM, NOJOB )
C
      CALL KSTAT ( SVALU,NOPNT,NOSIM,NSTEP,DSTEP,XRANG,YRANG,ANMOM,
     *             ANGLE,NOANG,NOJOB)
C
      IF (NOJOB.EQ.3) RETURN
  100 CONTINUE
      DO 80 IOSIM = 1,NPSIM
      CALL SPECT  ( NSTEP,NOPNT,XRANG,YRANG,COORD,IOSIM,DSTEP)
      CALL SMSTT  ( IOSIM, COORD, SVALU, ANMOM, NOJOB )
   80 CONTINUE
C
C     MAPPED PATTERN ANALYSIS
C
   85 CONTINUE
      DO 90 IOPNT = 1,NOPNT
      READ (8,*)  ( COORD(IOPNT,IDOFN),  IDOFN=1,NDOFN )
   90 CONTINUE
      IOSIM = NOSIM
      CALL SMSTT ( IOSIM, COORD, SVALU, ANMOM, NOJOB )
C
      CALL KSTAT  ( SVALU,NOPNT,NOSIM,NSTEP,DSTEP,XRANG,YRANG,ANMOM,
     *             ANGLE,NOANG,NOJOB )
      WRITE(6,910)
C
C     MONTE-CARLO STATISTIC
C
      WRITE(6,920)
      DO 70 IOSIM = 1,NOSIM
      CSTAT = 0.
      RSTAT = 0.
      DO 60 ISTEP = 1,NSTEP
      JSTAT = 0
      XSTAT = 0.
      ZSTAT = SVALU(IOSIM,ISTEP)
      DO 50 JOSIM = 1,NOSIM
      IF ( JOSIM.EQ.IOSIM ) GO TO 50
      XSTAT = XSTAT + SVALU(JOSIM,ISTEP)
   50 CONTINUE
      XSTAT = XSTAT / ( NOSIM - 1 )
      CSTAT = ( ZSTAT - XSTAT )**2
      RSTAT = RSTAT + CSTAT
   60 CONTINUE
      IF ( IOSIM.EQ.1 ) THEN
           RMINV = RSTAT
           RMAXV = RSTAT
      ENDIF
      IF ( IOSIM.NE.NOSIM.AND.RSTAT.LE.RMINV ) RMINV = RSTAT
      IF ( IOSIM.NE.NOSIM.AND.RSTAT.GE.RMAXV ) RMAXV = RSTAT
      WRITE(6,930) IOSIM, RSTAT
   70 CONTINUE
      WRITE(6,940) RMINV,RMAXV, RSTAT
C
  910 FORMAT(//,5X,'DOUBLY STOCHASTIC POINT PROCESS',// )
  920 FORMAT(//,5X,'*** MONTE - CARLO STATISTIC ***',//)
  930 FORMAT(   5X,'STATISTIC',I5,' = ',F15.3 )
  940 FORMAT(//,3X,'EXTREME VALUES OF SIMULATION',    /,
     *          5X,'MIN. VALUE OF STATISTIC = ',F15.3,/,
     *          5X,'MAX. VALUE OF STATISTIC = ',F15.3,/,
     *          5X,'MAPPED VALUE            = ',F15.3   )
C
      RETURN
      END
CC
C
      SUBROUTINE COXPP  ( NOPNT,XVALU,YVALU,VALUE,JITRT,XRANG,YRANG,
     *                    XBOT, YBOT  )
C
C     SUBROUTINE COXPP EVALUATE THE DOUBLY STOCHASTIC PROCESS
C     ( COX PROCESS ) IN A PLANE
C     INTENSITY FUNCTION IS ASSUMED TO SYMMETRIC NORMAL
C     DENSITY FUNCTION.
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION COORD(39,2), RSCLE(1000)
C
C     READ COORDINATES OF THE DATA AT THE FIRST ITERATION
C
      REWIND 8
      NOPN2 = NOPNT * 2
      PITWO = 6.283185
      IF ( JITRT.EQ.1) THEN
C
C     READ MEAN VALUE AND STANDARD DEVIATION OF THE NORMAL FUNCTION
C     LPOTN : 1 : INHOMOGENEOUS INTENSITY FUNCTION
C             2 : HOMOGENEOUS INTENSITY FUNCTION
C
```

```
         READ (8,*) VMEAN,SIGMA,LOPTN
C
         IF ( LOPTN.EQ.1 ) THEN
                 DO 10 IOPNT = 1,NOPNT
                 READ (8,*) ( COORD(IOPNT,IDOFN),IDOFN=1,2 )
   10            CONTINUE
         ELSE
                 DSEED = SECNDS(0.0) * 100.
                 CALL GGUBS ( DSEED,NOPN2,RSCLE )
                 DO 15 IDOFN = 1,NOPNT
                 JDOFN = IDOFN + NOPNT
                 COORD(IDOFN,1) = XBOT + RSCLE(IDOFN) * XRANG
                 COORD(IDOFN,2) = YBOT + RSCLE(JDOFN) * YRANG
   15            CONTINUE
         ENDIF
C
         WRITE(6,910) VMEAN,SIGMA
      ENDIF
C
C     AXI-SYMMETRIC BIVARIATE NORMAL DISTRIBUTION FUNCTION
C
      VALUE = 0.
      DO 20 IOPNT = 1,NOPNT
      XMEAN = COORD(IOPNT,1)
      YMEAN = COORD(IOPNT,2)
      XVAL1 = ( XVALU - XMEAN )**2
      YVAL1 = ( YVALU - YMEAN )**2
      FUNCT = EXP ( -0.5*( XVAL1 + YVAL1 ) / SIGMA**2 )
      VALUE = VALUE + FUNCT / ( PITWO*SIGMA**2 )
   20 CONTINUE
C
      VALUE = VALUE * VMEAN
C
  910 FORMAT(//,5X,'MEAN VALUE IN NORMAL FUNCTION = ',F5.3,/,
     *          5X,'STD. DEV.  IN NORMAL FUNCTION = ',F5.3,/)
      RETURN
      END
CC
C
      SUBROUTINE MAXCF ( COEMX, NOPNT, XRANG, YRANG, XBOT, YBOT )
C
C     CALCULATES THE MAX. DENSITY COEFFICIENT OF GIVEN DENSITY
C     FUNCTION
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION COEFT(5000)
C
      LXDIR = XRANG + 1
      LYDIR = YRANG + 1
      JITRT = 1
C
      READ (9,*) IXSTP,IYSTP
C
      COEMX = 0.
      DO 10 IXCON = 1, LXDIR, IXSTP
      DO 20 IYCON = 1, LYDIR, IYSTP
      XVALU = IXCON
      YVALU = IYCON
      CALL COXPP ( NOPNT,XVALU,YVALU,VALUE,JITRT,XRANG,YRANG,
     *             XBOT, YBOT  )
      COEFT(JITRT) = VALUE
C
C     FIND THE MAX. COEFFICIENT
C
      IF ( COEFT(JITRT).GT.COEMX ) THEN
          COEMX = COEFT(JITRT)
          IXREG = IXCON
          IYREG = IYCON
      ENDIF
      JITRT = JITRT + 1
   20 CONTINUE
   10 CONTINUE
C
      WRITE(6,910) COEMX
      WRITE(66,920) IXSTP,IYSTP,IXREG,IYREG,COEMX
  910 FORMAT(//,5X,'MAX. DENSITY COEFFICIENT = ',F10.4,//)
  920 FORMAT(//,5X,'X-DIR. STEP SIZE = ',I3,/,
     *          5X,'Y-DIR. STEP SIZE = ',I3,/,
     *          5X,'APPROX. X-COORD  = ',I3,/,
     *          5X,'APPROX. Y-COORD  = ',I3,/,
     *          5X,'MAX. DEN. COEFF. = ',F10.4,/)
C
      RETURN
      END
CC
C
      SUBROUTINE THINP ( NOGPT, HCOOR, COORD, NOPNT, COEMX, IOSIM,
     *                   XRANG, YRANG, XBOT,  YBOT  )
C
C     SUBROUTINE THINP EVALUATES THE THINNING PROCESS
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION HCOOR(5000,2), RSCLE(1), COORD(NOPNT,2)
C
C     CALL COXPP TO CALCULATE THE DENSITY COEFFICIENT
C
      KCOUN = 0
      JITRT = 1
   40 CONTINUE
      JITRT = JITRT + 1
      DO 10 IOGPT = 1,NOGPT
      XVALU = HCOOR(IOGPT,1)
      YVALU = HCOOR(IOGPT,2)
      IF ( XVALU.LT.0. ) GO TO 10
      CALL COXPP ( NOPNT,XVALU,YVALU,VALUE,JITRT,XRANG,YRANG,XBOT,YBOT )
C
C     CARE FOR A DSEED VALUE IN RADOP ROUTINE
C
      IF ( JITRT.EQ.2 ) THEN
           CALL RANDP ( 1, RSCLE, 0, IOGPT, 0 )
      ELSE
           CALL RANDP ( 1, RSCLE, 0, 0, 0 )
      ENDIF
C
      DENSE = VALUE / COEMX
      IF ( RSCLE(1).GT.DENSE ) GO TO 10
      KCOUN = KCOUN + 1
      DO 20 IDOFN = 1,2
      COORD(KCOUN,IDOFN)  = HCOOR(IOGPT,IDOFN)
   20 CONTINUE
      HCOOR(IOGPT,1)  = -HCOOR(IOGPT,1)
      IF ( KCOUN.EQ.NOPNT ) GO TO 30
   10 CONTINUE
   30 CONTINUE
C
C     CALCULATE THE EXACT NO. OF POINTS THAT SHOULD BE GENERATED
C
      WRITE(66,910)JITRT-1, KCOUN
      IF ( JITRT.GE.5 ) THEN
           WRITE(66,920) JITRT-1
           RETURN
      ENDIF
      IF ( KCOUN.NE.NOPNT ) GO TO 40
      IF ( IOSIM.EQ.1 ) THEN
         WRITE(6,930) KCOUN
         WRITE(6,940)
         DO 60 JOPNT = 1,KCOUN
         WRITE(6,950) JOPNT, (COORD(JOPNT,IDOFN),IDOFN=1,2)
   60    CONTINUE
      ENDIF
C
  910 FORMAT(//,5X,'NO. OF ITERATIONS TO GENERATE THE POINTS = ',I3,/,
     *          5X,'NO. POINTS GENERATED                     = ',I3,/)
  920 FORMAT(//,5X,'***  TOO MANY ITERATIONS (',I2,' )',//)
  930 FORMAT(//,5X,'NO. OF POINTS GENERATED = ',I5,///)
  940 FORMAT(//,5X,'DOUBLY STOCHASTIC POINT PATTERNS',/)
  950 FORMAT(3X,I3,5X,2(F15.7,3X))
      RETURN
      END
CC
C
      SUBROUTINE NORML ( NSTEP,NOPNT,XRANG,YRANG,COORD,IOSIM )
C
C     THE CORRELATION COEFFICIENTS WITH GGNML ( IMSL ).
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION COORD(106,2),RSCLE(1000)
C
      REWIND 10
      ICOUN = 1
      NOTPT = 0
      NOGOR = 1
      ROGOR = 1.
      SIGMA = 0.5
      SIGM1 = 1.
      NXSTP = XRANG / 2
      NYSTP = YRANG / 4
   60 CONTINUE
      NOTPT = 0
C
      DO 20 IYSTP = 1,NYSTP
      DO 10 IXSTP = 1,NXSTP
      IF ( IOSIM.EQ.1.AND.ICOUN.EQ.1 ) DSEED = 123457.D0
      ICOUN = 2
C
C     CALL NORMAL RANDOM DEVIATE GENERATOROF IMSL
C
      CALL GGNML ( DSEED,NOGOR,ROGOR )
      COEF1 = ROGOR*SQRT(0.5)
C
      CALL GGNML ( DSEED,NOGOR,ROGOR )
      COEF2 = ROGOR *SQRT(0.5)
C
C     CALCULATE THE INTENSITY FUNCTION WITH ABOVE COEFFICIENTS
C
      XLAM1 = NOPNT / ( XRANG * YRANG )
      XLAMB = XLAM1 + SIGMA**2 * COEF1 * COEF2
      IF ( XLAMB.LT.0. ) XLAMB = 0.
      XNOPT = XLAMB * 8.
      NOPT1 = XNOPT
      IF ( NOPT1.GT.0 ) THEN
          NOSUS = NOPT1 * 2
          CALL GGUBS ( DSEED,NOSUS,RSCLE )
          XBCOR = ( IXSTP - 1 ) * 2.
          YBCOR = ( IYSTP - 1 ) * 4.
          DO 40 ILOOP = 1,NOPT1
          IDOFN = ILOOP * 2 - 1
          JDOFN = ILOOP * 2
          IPOIN = NOTPT + ILOOP
          IF ( IPOIN.GT.NOPNT ) THEN
                 NOTPT = NOPNT
                 GO TO 30
          ENDIF
          COORD(IPOIN,1) = XBCOR + RSCLE(IDOFN) * 2
          COORD(IPOIN,2) = YBCOR + RSCLE(JDOFN) * 4.
   40     CONTINUE
          NOTPT = NOTPT + NOPT1
      ENDIF
      ICOUN = ICOUN + 1
   10 CONTINUE
   20 CONTINUE
C
   30 CONTINUE
      IF ( NOTPT.LT.NOPNT.OR.IYSTP.LT.(NYSTP-1) ) GO TO 60
      IF ( IOSIM.EQ.1 ) THEN
           WRITE(6,920)
           DO 50 IPOIN = 1,NOTPT
           WRITE(6,930) IPOIN, ( COORD(IPOIN,IDOFN),IDOFN=1,2 )
   50      CONTINUE
      ENDIF
```

```
C
  920 FORMAT(//,5X,'COORD. OF POINTS GENERATED BY COX PROCESS',/,
     *          5X,'NO.     X-COORD.     Y-COORD.',//)
  930 FORMAT(5X,I5,2(F10.3,3X))
      RETURN
      END
CC
C
      SUBROUTINE PAREN (COORD,IOSIM )
C
C     SUBROUTINE PAREN EVALUATE THE INTENSITY FUNCTION
C     OF THE COX PROCESS USING SYMMETRIC NORMAL
C     DISTRIBUTION FUNCTION FOR PARENT - DAUGHTER MODEL.
C     ( CURRENTLY, NOT AVAILABLE )
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONTR/NDOFN, DSTEP, NTOTL, NOPNT, NOSIM, NSTEP,
     *             NTOVL, IMAPP, IOPNT, XBOT,  XRANG, YBOT,
     *             YRANG, KXCLS, KYCLS, KTCLS, ANGLE, NOANG
      DIMENSION COORD(NOPNT,2), RSCLE(80)
      DIMENSION HCOOR(10,2),VALUE(50,40)
C
C     HCOOR(i,j)  : i = total no. of parent points
C                 : j = ndofn
C
C     VALUE(i,j)  : i = no. of simulations
C                 : j = no. of steps
C
      TWOPI = 6.28318
      XRAN1 = 14.
      YRAN1 = 14.
C
      IF ( IOSIM.EQ.1) READ (5,*) NOPAR, NOSON, STDEV, DISTL,PADLL
C
C     CALCULATE THE INTENSITIES WITH POISSON DISTRIBUTION
C     OF THE DATA
C
      IF ( IOSIM.EQ.1 ) READ (5,*) DSEEP, MAITN
    1 CONTINUE
      JPOIN = 0
      MODPT = 0
      NODAU = 2
      NOITN = 0
C
   10 CALL GGUBS ( DSEEP, NODAU, RSCLE )
      NOITN = NOITN + 1
      IF ( NOITN.GT.MAITN ) GO TO 1
C
C     GENERATE THE PARENT POINTS
C
      HCORX = RSCLE(1)  * XRAN1
      HCORY = RSCLE(2)  * YRAN1
      HCOXX = HCORX * 1.5 + 5.5
      HCOYY = HCORY - 5.5
      HCOR1 = 0.76 * HCOXX - 0.64 * HCOYY
      HCOR2 = 0.64 * HCOXX + 0.76 * HCOYY
      IF ( HCOR1.LE.XBOT.OR.HCOR1.GE.XRANG ) GO TO 10
      IF ( HCOR2.LE.YBOT.OR.HCOR2.GE.YRANG ) GO TO 10
C
      IF ( MODPT.GT.0 ) THEN
         DO 100 I = 1, MODPT
         PADIS = SQRT( (HCORX-HCOOR(I,1))**2+(HCORY-HCOOR(I,2))**2 )
         IF ( PADIS.LE.PADLL ) GO TO 10
  100    CONTINUE
      ENDIF
C
      MODPT = MODPT + 1
      HCOOR(MODPT,1) = HCORX
      HCOOR(MODPT,2) = HCORY
      IF ( MODPT.LT.NOPAR ) GO TO 10
C
C     GENERATE THE DAUGHTER POINTS
C
   20 CONTINUE
      CALL GGUBS ( DSEEP, 2, RSCLE )
      COORX = RSCLE(1) * XRAN1
      COORY = RSCLE(2) * YRAN1
      CORXX = COORX * 1.5 + 5.5
      CORYY = COORY - 5.5
      CORX2 = 0.76 * CORXX - 0.64 * CORYY
      CORY2 = 0.64 * CORXX + 0.76 * CORYY
      IF ( CORX2.LE.XBOT.OR.CORX2.GE.XRANG ) GO TO 20
      IF ( CORY2.LE.YBOT.OR.CORY2.GE.YRANG ) GO TO 20
      DO 30 IPOIN = 1, NOPAR
      DSTRI = SQRT( (COORX-HCOOR(IPOIN,1))**2 +
     *              (COORY-HCOOR(IPOIN,2))**2 )
      IF ( IPOIN.EQ.1 ) DSMIN = DSTRI
      IF ( DSTRI.LT.DSMIN ) DSMIN = DSTRI
   30 CONTINUE
      IF ( DSMIN.GT.DISTL ) GO TO 20
C
      JPOIN = JPOIN + 1
      COORD(JPOIN,1) = COORX
      COORD(JPOIN,2) = COORY
      IF ( JPOIN.LT.NOSON ) GO TO 20
      WRITE(6,920)
C
C     CONSTRUCT THE BIVARIARE NORMAL DISTRIBUTION FUNCTION
C
      DO 40 ISTEP = 1, NSTEP
      VALUE(IOSIM,ISTEP) = 0.
   40 CONTINUE
      DO 60 IPONT = 1, NOSON
      DO 50 JPONT = 1, NOPAR
      DISTS = SQRT( ( COORD(IPONT,1)-HCOOR(JPONT,1) )**2 +
     *              ( COORD(IPONT,2)-HCOOR(JPONT,2) )**2 )
C
      ISTEP = DISTS / DSTEP
      ISTEP = ISTEP + 1
      IF ( ISTEP.GT.NSTEP ) ISTEP = NSTEP
      XMEAN = HCOOR(JPONT,1)
      YMEAN = HCOOR(JPONT,2)
      XVAL1 = ( COORD(IPONT,1) - XMEAN )**2
      YVAL1 = ( COORD(IPONT,2) - YMEAN )**2
      FUNCT = EXP( -0.5*( XVAL1 + YVAL1 ) / STDEV**2 )
      VALFT = FUNCT / ( TWOPI * STDEV**2 )
      VALUE(IOSIM,ISTEP) = VALUE(IOSIM,ISTEP) + VALFT
   50 CONTINUE
   60 CONTINUE
C
      IF ( IOSIM.EQ.1) THEN
        DO 70 ISTEP = 1,NSTEP
        DSTAS = ISTEP * DSTEP
        WRITE(6,910) IOSIM,DSTAS,VALUE(IOSIM,ISTEP)
   70   CONTINUE
      ENDIF
      DO 80 I = 1,NOPAR
      J = I + NOSON
      COORD(J,1) = HCOOR(I,1)
      COORD(J,2) = HCOOR(I,2)
   80 CONTINUE
      WRITE(6,920)
      DO 90 I=1,NOPNT
      CORX3 = COORD(I,1) * 1.5 + 5.5
      CORY3 = COORD(I,2) - 5.5
      CORX4 = CORX3 * 0.76 - CORY3 * 0.64
      CORY4 = CORX3 * 0.64 + CORY3 * 0.76
      COORD(I,1) = CORX4
      COORD(I,2) = CORY4
      IF ( IOSIM.LE.5 ) THEN
         WRITE(6,960) I, (COORD(I,J),J=1,2)
      ENDIF
   90 CONTINUE
C
      WRITE(6,920)
C
  910 FORMAT(5X,'SIM. = ',I5,5X,2(F15.7,5X) )
  920 FORMAT(//)
  960 FORMAT(5X,I5,5X,2(F15.7,3X) )
C
      RETURN
      END
CC
C
      SUBROUTINE SPECT ( NSTEP,NOPNT,XRANG,YRANG,COORD,IOSIM,DSTEP )
C
C     SUBROUTINE SPECT CALCULATES THE SPECTRAL DENSITY FUNCTION
C     AND DIRECTIONAL INTENSITY FUNCTION.
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION COORD(39,2),RSCLE(60000),COOR1(30000,2),
     *          XINTE(200),  YINTE(400)
C
      IF ( IOSIM.EQ.1 ) THEN
           REWIND 8
           READ(8,*) XCOF2, YCOF2, FMULT, XVARI, YVARI,IPRIN
      ENDIF
      COEFF = 3.14159
      FREQL = COEFF / 2.
C
      LYLEN =50
C
      SUMMX = 0.
      SUMMY = 0.
      MCOUN = 0
      MMCNT = 0
      NOTPT = 0
C
C     XRAN1 & YRAN1 ARE THE DIRECTIONAL RANGES OF THE MAP
C     SHOULD BE CHANGED ACCORDING TO A SPECIFIC MAP CONCERNED
C
      XRAN1 = 22.
      YRAN1 = 14.
      IF ( IOSIM.EQ.1 ) JCUNT = 0
      XLAM1 = NOPNT / ( XRAN1 * YRAN1)
      IF ( IOSIM.GT.1 ) GO TO 100
C
C     X & Y-DIRECTIONAL SPECTRAL DENSITY FUNCTION
C
C     X-COOR & Y-COOR
C
      DELOM = COEFF / LYLEN
C
      DO 20 IXSTP = 1,10
      XCOOR =  FLOAT(IXSTP-1) * 3. + 1.5
C
      MMCNT = MMCNT + 1
C
      SUMMX = 0.
C
      DO 10 KSTEP = 0,LYLEN
      MCOUN = MCOUN + 1
      OMEGA = -FREQL + KSTEP*DELOM
      XSPEC = XCOF2 / ( COEFF * ( OMEGA**2 + XCOF2**2 ) )
      INUMB = 1
      IF ( IOSIM.EQ.1.AND.MCOUN.EQ.1 ) READ (8,*) DSEED
      CALL GGUBS ( DSEED, INUMB, RSCLE )
      THET1 = RSCLE(1) * 2. * COEFF
      XOEF5 = SQRT(2.*XSPEC*DELOM) * COS(OMEGA*XCOOR + THET1 )
      SUMMX = SUMMX + XOEF5
   10 CONTINUE
C
      XDENS = SUMMX * XVARI
      XINTE(IXSTP) = XDENS
      IF ( IXSTP.EQ.1 ) XDMAX = XDENS
      IF ( IXSTP.EQ.1 ) MAXXD = 1
      IF ( XDENS.GT.XDMAX ) THEN
           XDMAX = XDENS
           MAXXD = IXSTP
      ENDIF
```

```
C
   20 CONTINUE
C
      DO 40 IYSTP = 1,7
      YCOOR = FLOAT(IYSTP-1)*3.  + 1.5
      SUMMY = 0.
C
      DO 30 KSTEP = 0,LYLEN
      OMEGA = -FREQL + KSTEP*DELOM
      YSPEC = YCOF2 / ( COEFF * ( OMEGA**2 + YCOF2**2 ) )
      INUMB = 1
      CALL GGUBS ( DSEED, INUMB, RSCLE )
      THET2 = RSCLE(1) * 2. * COEFF
      YOEF5 = SQRT(2.*YSPEC*DELOM) * COS(OMEGA*YCOOR + THET2 )
      SUMMY = SUMMY + YOEF5
   30 CONTINUE
C
      YDENS = SUMMY * YVARI
      YINTE(IYSTP) = YDENS
      IF ( IYSTP.EQ.1 ) YDMAX = YDENS
      IF ( IYSTP.EQ.1 ) MAXYD = 1
      IF ( YDENS.GT.YDMAX ) THEN
           YDMAX = YDENS
           MAXYD = IYSTP
      ENDIF
C
   40 CONTINUE
      DO 700 IISTP = 1, 15
      CORNX = (IISTP-1.) * 3. + 1.5
      CORNY = (IISTP-1.) * 3. + 1.5
      XINT1 = XINTE(IISTP) / XDMAX
      YINT1 = YINTE(IISTP) / YDMAX
      WRITE(6,710) IISTP, CORNX,CORNY,
     * XINT1, YINT1
  700 CONTINUE
  710 FORMAT(5X,'STEP COORD X- Y- ',I5,4F10.4)
C
      DENMX  = ( XDMAX + YDMAX ) * FMULT
      DENMX1 = XDMAX + YDMAX
C
      READ(9,*) DENCR
      WRITE(6,940) XDMAX,MAXXD,YDMAX,MAXYD
  100 CONTINUE
      NOGPT = XRAN1 * YRAN1 * DENMX
      NOGP2 = NOGPT * 2
      IF ( IOSIM.EQ.1 ) READ (8,*) DSEE1
      CALL GGUBS ( DSEE1,NOGP2,RSCLE )
      JOGPT = 1
      DO 110 IOGPT = 1, NOGPT
      IDOFN = IOGPT * 2 - 1
      JDOFN = IOGPT * 2
C
      CORX1 = RSCLE(IDOFN) * XRAN1 + 5.5
      CORY1 = RSCLE(JDOFN) * YRAN1 - 5.5
      CORX2 = 0.76 * CORX1 - 0.64 * CORY1
      CORY2 = 0.64 * CORX1 + 0.76 * CORY1
      IF ( CORX2.LE.0..OR.CORX2.GE.XRANG ) GO TO 110
      IF ( CORY2.LE.0..OR.CORY2.GE.YRANG ) GO TO 110
      COOR1(JOGPT,1) = RSCLE(IDOFN)  * XRAN1
      COOR1(JOGPT,2) = RSCLE(JDOFN)  * YRAN1
      JOGPT = JOGPT + 1
  110 CONTINUE
      WRITE(6,950) JOGPT - 1
C
C     THINNING PROCESS
C
      DENST = 39. / ( 25. * 17. )
      JNUMB = 2
      DO 120 IOGPT = 1,JOGPT
      XTRI1 = COOR1(IOGPT,1)
      YTRI1 = COOR1(IOGPT,2)
      XADD1 = SQRT( XTRI1**2 + YTRI1**2 )
C
      IXADD = XADD1 * 0.33
      IXADD = IXADD + 1
      IYADD = YTRI1 * 0.333
      IYADD = IYADD + 1
      XXDEN = XINTE(IXADD)
      YYDEN = YINTE(IYADD)
      IF ( YYDEN.LE.0..OR.XXDEN.LE.0. ) GO TO 120
      XXDEN =  XXDEN /  XDMAX
      YYDEN =  YYDEN /  YDMAX
      XYDEN =  XXDEN + YYDEN
C
      CALL GGUBS ( DSEE1, JNUMB, RSCLE )
C
      IF ( XYDEN.GE.DENCR ) THEN
           NOTPT = NOTPT + 1
           IF ( NOTPT.GT.NOPNT ) GO TO 150
           CX1 = COOR1(IOGPT,1) + 5.5
           CY1 = COOR1(IOGPT,2) - 5.5
           COORD(NOTPT,1) = CX1*0.76 - CY1*0.64
           COORD(NOTPT,2) = CX1*0.64 + CY1*0.76
      ENDIF
  120 CONTINUE
C
  150 CONTINUE
      WRITE(6,910) IOGPT
      IF ( IOSIM.LE.IPRIN ) THEN
           WRITE(6,920)
           DO 160 IOPNT = 1,NOPNT
           WRITE(6,930) IOPNT, ( COORD(IOPNT,IDOFN),IDOFN = 1,2 )
  160      CONTINUE
      ENDIF
C
  910 FORMAT(5X,' NO. OF ITERATIONS = ',I5)
  920 FORMAT(//,5X,'COORD. OF POINTS GENERATED BY COX PROCESS',/,
     *            5X,'NO.    X-COORD.       Y-COORD.',//)
  930 FORMAT(5X,I5,2(F10.3,3X) )
  940 FORMAT(//,5X,'MAX. X-DIR. INTENSITY = ',F12.5,' AT ',I5,
     *        /,5X,'MAX. Y-DIR. INTENSITY = ',F12.5,' AT ',I5,//)
 950  FORMAT(//,5X,'TOTAL NO. OF POINTS GENERATED IN SPECT. = ',I5)
      RETURN
      END

      PROGRAM HIERA
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                            C
C     PROGRAM HIERA EVALUATES THE HIERARCHICAL FIBER         C
C     ( LINE - SEGMENT ) PROCESS WHICH IS A REALIZATION      C
C     OF THE GIVEN MAP.                                      C
C                                                            C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION KK(20000), AA(30000), TITLE(20)
C
C     INITIAL SETS OF THE DIMENSION
C
      NRVAR = 30000
      NIVAR = 20000
      NDOFN = 2
      NDOF2 = NDOFN * 2
      DFACT = 0.
C
      READ (5,800) TITLE
      WRITE(6,800) TITLE
C
   10 CONTINUE
C
C     READ VARIABLES
C
C     NOPNT : NO. OF TOTAL MID POINTS IN A MAP
C     NOFPT : NO. OF TOTAL MID POINTS IN A FORMER SET
C     NOLPT : NO. OF TOTAL MID POINTS IN A LATTER SET
C     NOSIM : NO. OF SIMULATIONS
C     NSTEP : NO. OF STEPS IN CALCULATING THE STATISTIC
C     DSTEP : DISTANCE STEP SIZE
C     SIGMA : STD. DEV. OF THE LINE - KERNEL FUNCTION
C     TSTEP : UNIT OF SEGMENT IN LINE - KERNEL FUNCTION
C             OR THE PERTURBATION DISTANCE IN INDEPENDENCE TEST
C
      READ (5,*) NOPNT,NOFPT,NOLPT,NOSIM,NSTEP,DSTEP,SIGMA,TSTEP
      WRITE(6,910) NOPNT,NOFPT,NOLPT,NOSIM,NSTEP,DSTEP,SIGMA,TSTEP
C
C     READ BOUNDARY OF THE SIMULATION REGION
C
      READ (5,*) XBOT,XRANG,YBOT,YRANG
      XTOP = XBOT + XRANG
      YTOP = YBOT + YRANG
      WRITE(6,920) XBOT,XTOP,YBOT,YTOP
C
      MAXXW = 2
      MAXYW = 2
      READ (5,*) IOPTN, JOPTN, LOPTN
      IF ( JOPTN.GE.3 ) THEN
           READ (5,*) XWIND,YWIND,DFACT
           MAXXW = XRANG / XWIND + 1
           MAXYW = YRANG / YWIND + 1
      ENDIF
C
C     DYNAMIC ALLOCATION
C
C       NR1 : COORD  :  COORD. OF THE TOTAL TRACES ( 4 FOR EACH TRACE )
C       NR2 = CORFB  :  COORD. OF THE FORMER SET   (         "        )
C       NR3 = SECON  :  SLOPE OF ALL FIBERS        ( NO. OF FIBERS )
C       NR4 = TRACE  :  TRACE LENGTH OF THE FORMER SET
C       NR5 = CORLF  :  COORD. OF THE LATTER SET   (         "        )
C       NR6 = COORL  :  WORKING ARRAY FOR THE LATTER SET  (2 FOR TRACE)
C       NR7 = COORF  :  WORKING ARRAY FOR THE LATTER SET  (     "     )
C       NR8 = SVALU  :  VALUES OF THE STATISTIC ( NOSIM,NSTEP )
C       NR9 = CPFST  :  CENTER POINTS OF THE WHOLE SET ( NOPNT,2 )
C       NR10= CPSND  :  CENTER POINTS OF LATTER SET    ( NOLPT,2 )
C       NR11= DENTY  :  INTENSITY VARIATION OF LATTER SET ( JOPTN=2 )
C
      NTOTL = NDOFN * NOPNT
      NTOT2 = NDOF2 * NOPNT
      NTFPT = NDOF2 * NOFPT
      NTLPT = NDOF2 * NOLPT
      NOSAS = NOSIM * NSTEP
      NOWIN = MAXXW * MAXYW
C
      NR1   =   1
      NR2   = NR1  +   NTOT2
      NR3   = NR2  +   NTFPT
      NR4   = NR3  +   NOPNT
      NR5   = NR4  +   NOFPT
      NR6   = NR5  +   NTLPT
      NR7   = NR6  +   NOLPT * 2
      NR8   = NR7  +   NOFPT * 2
      NR9   = NR8  +   NOSAS
      NR10  = NR9  +   NOPNT * 2
      NR11  = NR10 +   NOLPT * 2
      NR12  = NR11 +   NOWIN
C
      NI1   =   1
C
      NRTOT = NR12 - 1
      NITOT = NI1 - 1
      IERRO = 0
C
      WRITE(6,930) NRTOT, NRVAR
      IF ( NRVAR.GT.NRTOT ) GO TO 20
      WRITE(6,940)
      IERRO = IERRO + 1
   20 CONTINUE
      WRITE(6,950) NITOT, NIVAR
      IF ( NIVAR.GT.NITOT ) GO TO 30
      WRITE(6,960)
      IERRO = IERRO + 1
   30 CONTINUE
      IF ( IERRO.GT.0 ) STOP
C
C     INITIALIZE
C
      DO 40 IVARI = 1,NRVAR
   40 AA(IVARI) = 0.
      DO 50 IVARI = 1,NIVAR
   50 KK(IVARI) = 0
C
C     CALL MAIN
C
      CALL MAINS ( AA(NR1 ), AA(NR2 ), AA(NR3 ), AA(NR4 ), AA(NR5 ),
     *             AA(NR6 ), AA(NR7 ), AA(NR8 ), AA(NR9 ), AA(NR10),
     *             AA(NR11)  )
C
  800 FORMAT(20A4)
  910 FORMAT(//,5X,'NO. OF TOTAL MID POINTS IN A MAP        =',T60,I5,
     *        /,5X,'NO. OF TOTAL MID POINTS IN A FORMER SET =',T60,I5,
     *        /,5X,'NO. OF TOTAL MID POINTS IN A LATTER SET =',T60,I5,
     *        /,5X,'NO. OF SIMULATIONS                      =',T60,I5,
     *        /,5X,'NO. OF INTERPRETATION STEPS             =',T60,I5,
     *        /,5X,'DISTANCE STEP SIZE                      =',T60,F6.3,
     *        /,5X,'STD. DEV. OF LINE-KERNEL FUNCTION       =',T60,F6.3,
     *        /,5X,'UNIT OF SEGMENT IN LINE-KERNEL FUNCTION =',T60,F6.3)
  920 FORMAT(//,5X,'BOUNDARY OF THE SIMULAITON',
     *        /,7X,'X-DIR. BOUNDARY ( FROM : TO ) = ',T60,2(F10.3,2X),
     *        /,7X,'Y-DIR. BOUNDARY ( FROM : TO ) = ',T60,2(F10.3,2X) )
  930 FORMAT(//,5X,'REAL STORAGE REQUIRED       = ',I5,
     *        /,5X,'REAL STORAGE SPECIFIED      = ',I5,/)
  940 FORMAT(//,5X,'*** INCREASE STORAGE FOR REAL ARRAYS ***')
  950 FORMAT(//,5X,'INTEGER STORAGE REQUIRED    = ',I5,
     *        /,5X,'INTEGER STORAGE SPECIFIED   = ',I5,/)
  960 FORMAT(//,5X,'*** INCREASE STORAGE FOR INTEGER ARRAYS ***')
C
      STOP
      END
CC
C
      SUBROUTINE MAINS ( COORD, CORFB, SECON, TRACE, CORLF, COORL,
     *                   COORF, SVALU, CPFST, CPSND, DENTY )
C
C     SUBROUTINE MAINS CONTROLS THE MAIN OPTION
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION COORD(NOPNT,4), CORFB(NOFPT,4), SECON(NOPNT),
     *          CORLF(NOLPT,4), COORL(NOLPT,2), TRACE(NOFPT),
     *          COORF(NOFPT,2), CPFST(NOPNT,2), CPSND(NOLPT,2)
      DIMENSION SVALU(NOSIM,NSTEP), DENTY(MAXXW,MAXYW)
C
```

```
C     READ OPTION VARIABLES
C
C       IOPTN : MAIN OPTION FOR THE POINT PROCESS
C         ( 0 : DO NOT ENTER THE POINT PROCESS   )
C         ( 1 : BIVARIATE KERNEL FUNCTION METHOD )
C         ( 2 : INDEPENDENCE TEST FOR SETS       )
C         ( 3 : ORIENTATION CORRELATION OPTION   )
C         ( 4 : MLE FOR ORIENTATION DATA         )
C
C       JOPTN : MAIN OPTION FOR THE FIBER PROCESS
C         ( 0 : DO NOT ENTER THE FIBER PROCESS  )
C         ( 1 : MLE FOR LINE - KERNEL FUNCTION  )
C         ( 2 : MLE FOR NEAREST NEIGHBOR FIBERS )
C         ( 3 : SIMULATION OF SET 2 WITH L - K  )
C         ( 4 : SIMULATION OF SET 2 WITH NN F   )
C
C       LOPTN : ANALYSIS OPTION
C         ( 1 : CALCULATE BIVARIATE  SECOND MOMENT )
C         ( 2 : CALCULATE UNIVARIATE SECOND MOMENT )
C
      WRITE(6,910) IOPTN,JOPTN,LOPTN
C
C     FIRST, SIMULATE THE FORMER SET USING POINT PROCESS
C
      IF ( IOPTN.EQ.0 ) GO TO 99
      GO TO ( 10, 20, 30, 30 ), IOPTN
C
C     IOPTN = 1 : BIVARIATE KERNEL FUNCTION METHOD
C
   10 CONTINUE
      GO TO 90
C
C     IOPTN = 2 : BIVARIATE INDEPENDENCE TEST
C
   20 CONTINUE
      CALL INDEP ( COORD,CORFB,CORLF,SVALU,CPFST,CPSND )
      GO TO 90
C
C     IOPTN = 3 : ORIENTATION CORRELATION OPTION ( BIVARIATE )
C             4 : MLE FOR ORIENTATION DATA
C
   30 CONTINUE
      CALL ORINT ( COORD, TRACE, SECON )
   90 CONTINUE
C
C        ( 1 : LOG DISTRIBUTION           )
C        ( 2 : EXPONENTIAL DISTRIBUTION   )
C
      CALL LENGT ( KOPTN, COORF )
   99 CONTINUE
C
      IF ( JOPTN.EQ.0 ) RETURN
C
C     SECOND, SIMULATE THE LATTER SET USING FIBER PROCESS
C
```

```
      IF ( JOPTN.LE.2 ) READ (5,*) XLAMO
      DO 60 IOPNT = 1, NOFPT
      READ (5,*) ( COORD(IOPNT,IDOFN), IDOFN = 1, NDOF2 )
   60 CONTINUE
C
      GO TO ( 100, 200, 300, 300 ), JOPTN
C
C     JOPTN = 1 : MLE OF LINE - KERNEL FUNCTION
C
  100 CONTINUE
      CALL KFMLE ( COORD,CORFB,SECON,TRACE,CORLF,COORL,SVALU,
     *             XLAMO )
      RETURN
C
C     JOPTN = 2 : MLE OF NEAREST NEIGHBOR FIBERS
C
  200 CONTINUE
      CALL NNMLE ( COORD, TRACE, SECON, XLAMO )
      RETURN
C
C      JOPTN = 3 : SIMULATION OF SET 2 WITH L - K FUNCTION
C
C      JOPTN = 4 : SIMULATION OF SET 2 WITH N.N. FUNCTION
C
  300 CONTINUE
      CALL FIBRE ( COORD,CPFST,SECON,TRACE, CPSND, COORL, SVALU,
     *             DENTY )
      RETURN
C
  910 FORMAT(//,5X,'MAIN OPTIONS :',
     *        /,7X,'MAIN OPTION FOR THE POINT PROCESS       = ',T60,I5,
     *        /,9X,'( 0 : DO NOT ENTER THE POINT PROCESS   )',
     *        /,9X,'( 1 : BIVARIATE KERNEL FUNCTION METHOD )',
     *        /,9X,'( 2 : INDEPENDENT TEST FOR SETS        )',
     *        /,9X,'( 3 : CORRELATION MEASURE OF ANGLE     )',
     *        /,9X,'( 4 : MLE OF ORIENTATION DATA          )',
     *       //,7X,'MAIN OPTION FOR THE FIBER PROCESS       = ',T60,I5,
     *        /,9X,'( 0 : NOT CONSIDER THE FIBER PROCESS   )',
     *        /,9X,'( 1 : MLE OF LINE - KERNEL FUNCTION    )',
     *        /,9X,'( 2 : MLE OF NEAREST NEIGHBOR FIBERS   )',
     *        /,9X,'( 3 : SIMULATION WITH L - K FUNCTION   )',
     *        /,9X,'( 4 : SIMULATION WITH N. N. FUNCTION   )',
     *       //,7X,'ANALYSIS OPTION FOR POINT PATTERN       = ',T60,I5,
     *        /,9X,'( 1 : BIVARIATE  SECOND MOMENT OPTION  )',
     *        /,9X,'( 2 : UNIVARIATE SECOND MOMENT OPTION  )' )
C
      END
CC
C
      SUBROUTINE FIBRE ( COORD,CPFST,SECON,TRACE,CPSND,COORL,SVALU,
     *                   DENTY )
C
C     SUBROUTINE FIBRE EVALUATES THE HIERARCHICAL FIBER MODEL
C     WITH LINE - KERNEL FUNCTION.
C
```

```
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *              NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *              MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION COORD(NOPNT,4), CPFST(NOPNT,2), SECON(NOPNT),
     *          CPSND(NOLPT,2), COORL(NOLPT,2), TRACE(NOFPT)
      DIMENSION SVALU(NOSIM,NSTEP), RSCLE(2)
      DIMENSION COORW(138,2), DENTY(MAXXW,MAXYW)
C
C     SIMULATE THE SECOND SET USING THE DATA FROM MLE
C
      JITRT = 1
      TWOPI = 6.28318
      NPLU1 = NOFPT + 1
      NPLU2 = NOFPT + NOLPT
      XLAM2 = NOLPT / ( XRANG * YRANG )
C
C     STORE THE MID POINTS DATA FOR SECOND MOMENT ANALYSIS
C
      DO 5 IOPNT = 1, NOFPT
      XCOOR = ( COORD(IOPNT,1) + COORD(IOPNT,3) ) / 2.
      YCOOR = ( COORD(IOPNT,2) + COORD(IOPNT,4) ) / 2.
      CPFST(IOPNT,1) = XCOOR
      CPFST(IOPNT,2) = YCOOR
    5 CONTINUE
C
C     READ MID POINT OF THE LATTER SET ( MAPPED DATA )
C
      DO 10 IOPNT = NPLU1, NPLU2
      READ (5,*) ( COORD(IOPNT,IDOFN), IDOFN=1,NDOF2)
      XCOOR = ( COORD(IOPNT,1) + COORD(IOPNT,3) ) / 2.
      YCOOR = ( COORD(IOPNT,2) + COORD(IOPNT,4) ) / 2.
      JOPNT = IOPNT - NOFPT
      COORW(JOPNT,1) = XCOOR
      COORW(JOPNT,2) = YCOOR
   10 CONTINUE
C
C     FIND INTENSITY VARIATION WITH BIVARIATE NORMAL DENSITY FUNCTION
C
C      MAXXW = XRANG / XWIND + 1
C      MAXYW = YRANG / YWIND + 1
C
      XWIND = XRANG / ( MAXXW - 1. )
      YWIND = YRANG / ( MAXYW - 1. )
      WRITE(6,950) XWIND, YWIND, DFACT
C
      DO 20 IXWIN = 1, MAXXW
      DO 20 IYWIN = 1, MAXYW
      DENTY(IXWIN,IYWIN) = 0.
   20 CONTINUE
C
      JITRT = 1
      DO 30 IOPNT = 1, NOLPT
      XCOOR = COORW(IOPNT,1)
      YCOOR = COORW(IOPNT,2)
```

```
C
      IXWIN = XCOOR / XWIND + 1
      IYWIN = YCOOR / YWIND + 1
C
C     JOPTN = 3 : SIMULATION WITH L - K FUNCTION
C
      IF ( JOPTN.EQ.3 ) THEN
         CALL LKERN  (JITRT,XCOOR,YCOOR,TRACE,VALUE,COORD,SECON,
     *               NTOFB)
         JITRT = 2
         DENTY(IXWIN,IYWIN) = DENTY(IXWIN,IYWIN) +
     *          VALUE * 0.35 / FLOAT(NOLPT) + 0.65 / (22.*14.)
      ELSE
C
C     JOPTN = 4 : SIMULATION WITH N. N. FUNCTION
C
         CALL NEARF  (JITRT,XCOOR,YCOOR,FDIST,COORD,TRACE,SECON)
         JITRT = 2
         DENTY(IXWIN,IYWIN) = DENTY(IXWIN,IYWIN) +
     *          EXP( -0.5 * (FDIST/SIGMA)**2 ) / ( SIGMA*SQRT(TWOPI)
     *        * FLOAT(NOLPT) ) * 0.35 + 0.65 / (22.*14.)
      ENDIF
C
      IF ( IOPNT.EQ.1 ) DENMX = DENTY(IXWIN,IYWIN)
      IF ( DENTY(IXWIN,IYWIN).GT.DENMX ) DENMX = DENTY(IXWIN,IYWIN)
   30 CONTINUE
C
      DENMX = DENMX * DFACT
      KOPTN = 1
C
C     SIMULATION STEP
C
      NPSIM = NOSIM - 1
      IF ( NPSIM.EQ.0 ) RETURN
C
      DO 50 IOSIM = 1, NPSIM
      IF ( IOSIM.LE.5 ) WRITE(6,910) IOSIM
C
C     SIMULATION BY NORMAL DISTRIBUTION FUNCTION
C
      ICONT = 0
      KCONT = 0
C
  220 CONTINUE
      KCONT = KCONT + 1
C
C     GENERATE RANDOM POINT WHICH REPRESENTS THE MID POINT OF THE
C       LATTER SET.
C
      NOSUS = 2
      CALL RANDF ( NOSUS,RSCLE,KOPTN )
      KOPTN = 2
C
      XCOOR = RSCLE(1)  * XRANG
      YCOOR = RSCLE(2)  * YRANG
      IXWIN = XCOOR / XWIND
      IYWIN = YCOOR / YWIND
      IXWIN = IXWIN + 1
      IYWIN = IYWIN + 1
      VALUE = DENTY(IXWIN,IYWIN)
C
      CALL RANDF ( 1,RSCLE,KOPTN )
      IF ( KCONT.GT.10000 ) RETURN
      IF ( RSCLE(1).GT. (VALUE/DENMX) ) GO TO 220
      ICONT = ICONT + 1
      COORL(ICONT,1) = XCOOR
      COORL(ICONT,2) = YCOOR
C
      IF (IOSIM.LE.5) WRITE(6,920) ICONT, COORL(ICONT,1), COORL(ICONT,2)
      IF ( ICONT.LT.NOLPT ) GO TO 220
      WRITE(6,930) KCONT
C
C     CALCULATE APPROPRIATE SECOND MOMENT
C       LOPTN = 1 : BIVARIATE  SECOND MOMENT OPTION
C               2 : UNIVARIATE SECOND MOMENT OPTION
C
      IF ( LOPTN.EQ.2 ) THEN
C
C     CALCULATE THE SECOND MOMENT
C
         CALL SMSTT ( IOSIM, COORL, SVALU )
      ELSE
C
C     CALCULATE THE BIVARIATE SECOND MOMENT
C
         DO 230 KOPNT = 1, NOFPT
         XCOOR = CPFST(KOPNT,1)
         YCOOR = CPFST(KOPNT,2)
         CALL BVSMM ( XCOOR, YCOOR, COORL, SVALU, IOSIM )
  230    CONTINUE
      ENDIF
C
   50 CONTINUE
C
C     CALCULATE  THE SECOND MOMENT FOR REAL DATA SET 2
C
      IOSIM = NOSIM
C
      IF ( LOPTN.EQ.2 ) THEN
           CALL SMSTT ( IOSIM, COORW, SVALU )
      ELSE
C
C     CALCULATE THE BIVARIATE SECOND MOMENT FOR REAL DATA SET 2
C
           DO 240 KOPNT = 1, NOFPT
           XCOOR = CPFST(KOPNT,1)
           YCOOR = CPFST(KOPNT,2)
           CALL BVSMM ( XCOOR, YCOOR, COORW, SVALU, IOSIM )
  240      CONTINUE
      ENDIF
```

```
C
C     CALCULATE THE CUMULATIVE K FUNCTION
C
      DO 260 IOSIM = 1, NOSIM
      DO 250 ISTEP = 1, NSTEP
      ADDTV = 0.
      IF ( ISTEP.EQ.1 ) GO TO 250
      KSTEP = ISTEP - 1
      ADDTV = SVALU(IOSIM,KSTEP)
      SVALU(IOSIM,ISTEP) = SVALU(IOSIM,ISTEP) + ADDTV
  250 CONTINUE
  260 CONTINUE
C
      CALL KSTAT ( SVALU )
C
C     MONTE - CARLO STATISTICS
C
      WRITE(6,890)
      DO 90 IOSIM = 1, NOSIM
      CSTAT = 0.
      RSTAT = 0.
      DO 80 ISTEP = 1, NSTEP
      JSTAT = 0
      XSTAT = 0.
      ZSTAT = SVALU(IOSIM,ISTEP)
      DO 70 JOSIM = 1, NOSIM
      IF ( JOSIM.EQ.IOSIM ) GO TO 70
      XSTAT = XSTAT + SVALU(JOSIM,ISTEP)
   70 CONTINUE
      XSTAT = XSTAT / ( NOSIM - 1 )
      CSTAT = ( ZSTAT - XSTAT  )**2
      RSTAT = RSTAT + CSTAT
   80 CONTINUE
      IF ( IOSIM.EQ.1 ) THEN
           RMINV = RSTAT
           RMAXV = RSTAT
      ENDIF
      IF ( IOSIM.NE.NOSIM.AND.RSTAT.LE.RMINV ) RMINV = RSTAT
      IF ( IOSIM.NE.NOSIM.AND.RSTAT.GE.RMAXV ) RMAXV = RSTAT
      WRITE(6,980) IOSIM, RSTAT
   90 CONTINUE
      WRITE(6,990) RMINV, RMAXV, RSTAT
C
  890 FORMAT(//,5X,'MONTE - CARLO TEST',//)
  910 FORMAT(//,5X,'SIMULATIONS OF THE SET 2 WITH LINE - KERNEL FT.',
     *        /,5X,' ITERATION STEP = ',I5,
     *       //,9X, 'X - COORD.                        Y - COORD',//)
  920 FORMAT(   5X,I5,3X,2(F15.7,3X) )
  930 FORMAT(/ ,5X,'NO. OF ITERATIONS = ',I5,//)
  950 FORMAT(//,5X,'X-DIR. WINDOW WIDTH = ',F10.5,
     *        /,5X,'Y-DIR. WINDOW WIDTH = ',F10.5,
     *        /,5X,'DENSITY FACTOR      = ',F10.5,
     *       // )
  980 FORMAT(   5X,'STATISTIC',I5,'    = ',F15.3 )
  990 FORMAT(//,5X,'MIN. VALUE OF STATISTIC  = ',F15.3,
     *        /,5X,'MAX. VALUE OF STATISTIC  = ',F15.3,
     *        /,5X,'MAPPED VALUE             = ',F15.3 )
C
      RETURN
      END
CC
C
      SUBROUTINE SMSTT ( IOSIM, COORL, SVALU )
C
C     SUBROUTINE SMSTT EVALUATE SECOND-MOMENT STATISTIC OF
C     MAPPED DATA
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION COORL(NOLPT,2), SVALU(NOSIM,NSTEP)
C
      NPSIM = NOSIM - 1
      CNSTV = 1.
C
      DO 30 ISTEP = 1,NSTEP
      SVALU(IOSIM,ISTEP) = 0.
   30 CONTINUE
C
C     CALCULATE WEIGHTING COEFFICIENT w(X,u)
C
      DO 20 IPOIN = 1, NOLPT
      XCOOR = COORL(IPOIN,1)
      YCOOR = COORL(IPOIN,2)
      DIST1 = AMIN1 ( XCOOR, (XRANG - XCOOR) )
      DIST2 = AMIN1 ( YCOOR, (YRANG - YCOOR) )
      SQDIS = DIST1**2 + DIST2**2
      DO 10 JPOIN = 1,NOLPT
      IF ( JPOIN.EQ.IPOIN ) GO TO 10
      XCOR1 = COORL(JPOIN,1)
      YCOR1 = COORL(JPOIN,2)
      DIST0 = SQRT((XCOOR-XCOR1)**2+(YCOOR-YCOR1)**2)
C
      WEIGT = 0.
      ISTEP = DIST0 / DSTEP + 1
      IF ( ISTEP.GT.NSTEP ) GO TO 10
      IF ( DIST0**2.LE.SQDIS ) THEN
              WEIGT = 1.
      ELSE
           DIST3 = ACOS( CNSTV * DIST1 / DIST0 )
           DIST4 = ACOS( CNSTV * DIST2 / DIST0 )
           WEIGT = 0.75 - ( DIST3 + DIST4 ) / 6.28318
      ENDIF
      WEIGT = XRANG * YRANG / ( WEIGT * (NOLPT)**2 )
C
C     CALCULATE K FUNCTION
C
      SVALU(IOSIM,ISTEP) = SVALU(IOSIM,ISTEP) + WEIGT
```

```
C
   10 CONTINUE
   20 CONTINUE
C
      RETURN
      END
CC
C
      SUBROUTINE KSTAT ( SVALU )
C
C     SUBROUTINE KSTST INTERPRET THE K FUNCTION
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION SVALU(NOSIM,NSTEP)
C
C     FIND MIN. & MAX. VALUE OF SIMULATED DATA
C
      AREAT = XRANG * YRANG
      XLAMB = NOLPT / AREAT
      XLAM2 = XLAMB**2
C
      WRITE(6,910)
C
      NPSIM = NOSIM - 1
      PHIVU = 3.141592
C
      DO 20 ISTEP = 1,NSTEP
      STOTL = 0.
      SMINV = SVALU(1,ISTEP)
      SMAXV = SVALU(1,ISTEP)
      STOTL = SVALU(1,ISTEP)
      DO 10 IOSIM = 2,NPSIM
      IF ( SVALU(IOSIM,ISTEP).LT.SMINV ) SMINV = SVALU(IOSIM,ISTEP)
      IF ( SVALU(IOSIM,ISTEP).GT.SMAXV ) SMAXV = SVALU(IOSIM,ISTEP)
      STOTL = STOTL + SVALU(IOSIM,ISTEP)
   10 CONTINUE
      SAVRG = STOTL / NPSIM
C
C     MAPPED PATTERN
C
   15 CONTINUE
C
      SMAPP = SVALU(NOSIM,ISTEP)
C
C     STABILIZE THE K VALUE
C
      DSTNS = DSTEP * ISTEP
      DSTS2 = DSTNS **2
      SMINV = SMINV - PHIVU*DSTS2
      SMAXV = SMAXV - PHIVU*DSTS2
      SAVRG = SAVRG - PHIVU*DSTS2
      SMAPP = SMAPP - PHIVU*DSTS2
C
      WRITE(6,920)DSTNS,SMINV,SMAXV,SAVRG,SMAPP
   20 CONTINUE
C
  910 FORMAT(//,7X,'INHOMOGENEOUS POISSON POINT PROCESS', /,
     *         /,7X,'SECOND-MOMENT MEASUREMENT STATISTIC',//,
     *            3X,'DISTANCE  MIN. VALUE  MAX. VALUE',
     *            2X,'AVERAGE  MAPPED    ' //)
  920 FORMAT(    4X,F9.3,3X,4(F12.4,2X) )
C
      RETURN
      END
CC
C
      SUBROUTINE LKERN ( JITRT,XCOOR,YCOOR,TRACE,VALUE,COORD,
     *                   SECON,NTOFB)
C
C     SUBROUTINE LKERN CALCULATE THE LINE KERNEL FUNCTION
C     OF THE FIBER PROCESS.
C     AS AN ASSUMPTION, WE ONLY CONSIDER THE LINE LENGTH.
C
C     IMPLICIT DOUBLE PRECISION ( A-H, O-Z )
C
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION COORD(NOPNT,4), TRACE(NOFPT), SECON(NOPNT)
C
      PITWO = 6.283185
      VALUE = 0.
C
      IF ( JITRT.EQ.1 ) THEN
C
C     READ END COORDINATES OF THE FORMER SET.
C       COORD(IOPNT,1) & COORD(IOPNT,2) : STARTING POINT
C       COORD(IOPNT,3) & COORD(IOPNT,4) : END POINT
C
        DO 10 IOPNT = 1, NOFPT
        IF ( COORD(IOPNT,1).GT.COORD(IOPNT,3) ) THEN
                CORXP = COORD(IOPNT,3)
                CORYP = COORD(IOPNT,4)
                COORD(IOPNT,3) = COORD(IOPNT,1)
                COORD(IOPNT,4) = COORD(IOPNT,2)
                COORD(IOPNT,1) = CORXP
                COORD(IOPNT,2) = CORYP
        ENDIF
C
C     CALCULATE THE LINE EQUATION FOR THE GIVEN FIBER
C       y = secod * ( x - Xi ) + Yi
C       trace = trace length
C
        SECOD = ( COORD(IOPNT,4) - COORD(IOPNT,2) ) /
     *          ( COORD(IOPNT,3) - COORD(IOPNT,1) )
        SECON(IOPNT) = SECOD
        TRACE(IOPNT) = SQRT( (COORD(IOPNT,3) - COORD(IOPNT,1))**2 +
     *                       (COORD(IOPNT,4) - COORD(IOPNT,2))**2 )
   10   CONTINUE
      ENDIF
C
C     LOOKING FOR A DISTANCE BETWEEN FIBER AND DATA POINT
C
      NTOFB = 0
      DO 20 IOPNT = 1, NOFPT
      FTLEN = TRACE(IOPNT)
      NITER = FTLEN / TSTEP
      IF ( NITER.EQ.0 ) NITER = 1
      NTOFB = NTOFB + NITER
C
      VALU1 = 0.
      DO 30 IITER = 1, NITER
      TLITN = TSTEP * ( IITER - 1 ) + TSTEP / 2.
      XTRPT = COORD(IOPNT,1) + TLITN * ( COORD(IOPNT,3) -
     *        COORD(IOPNT,1 ) ) / TRACE(IOPNT)
      YTRPT = COORD(IOPNT,2) + TLITN * ( COORD(IOPNT,4) -
     *        COORD(IOPNT,2) ) / TRACE(IOPNT)
      IF ( NITER.EQ.1 ) THEN
        XTRPT = ( COORD(IOPNT,3) + COORD(IOPNT,1) ) / 2.
        YTRPT = ( COORD(IOPNT,4) + COORD(IOPNT,2) ) / 2.
      ENDIF
C
      DTRPT = SQRT( (XCOOR-XTRPT)**2 + (YCOOR-YTRPT)**2 )
C
      VALU1 = VALU1 + EXP( -0.5 * (DTRPT/SIGMA)**2 ) /
     *                ( SIGMA **2 *  PITWO )
   30 CONTINUE
C
C     CALCULATE THE TOTAL INFLUENCE OF THE FIBER PROCESS
C
      VALUE = VALUE + VALU1
   20 CONTINUE
C
      RETURN
      END
CC
C
      SUBROUTINE RANDF ( NOSUS, RSCLE, KOPTN )
C
C     SUBROUTINE RANDF GENERATE PSEUDO-RANDOM NUMBER
C     USING IMSL LIBRARY
C
      DIMENSION RSCLE(NOSUS)
C
      IF ( KOPTN.EQ.1 ) DSEED = SECNDS(0.0) * 100.0
      CALL GGUBS ( DSEED, NOSUS, RSCLE )
C
      RETURN
      END
CC
C
```

```
      SUBROUTINE LENGT ( KOPTN, COORF )
C
      RETURN
      END
CC
C
      SUBROUTINE INDEP (COORD,CORFB,CORLF,SVALU,CPFST,CPSND)
C
C     SUROUTINE INDEP EVALUATE THE INDEPENDENCE TEST
C     FOR BIVARIATE POINT PROCESS
C     USING EITHER SMALL PERTURBAION METHOD OR TOROIDAL SHIFT SCHEME
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION COORD(NOPNT,4), CORFB(NOFPT,4), CORLF(NOLPT,4),
     *          SVALU(NOSIM,NSTEP)
      DIMENSION RSCLE(2)
      DIMENSION CPFST(177,2), CPSND(138,2)
C
C     INITIATE WITH THE ORIGINAL MID POINT DATA OF FORMER &
C     LATTER SETS
C
      NPLU1 = NOFPT + 1
      NPLU2 = NOFPT + NOLPT
C
      DO 10 IOPNT = 1, NOPNT
      READ (5,*) ( COORD(IOPNT,IDOFN), IDOFN=1,NDOF2 )
      XCOOR = ( COORD(IOPNT,1) + COORD(IOPNT,3) ) / 2.
      YCOOR = ( COORD(IOPNT,2) + COORD(IOPNT,4) ) / 2.
      CPFST(IOPNT,1) = XCOOR
      CPFST(IOPNT,2) = YCOOR
   10 CONTINUE
C
C     CALCULATE 2nd MOMENT OF BIVARIATE POINT PROCESS
C
C     SELECT ANALYSIS OPTION
C       ISHFT = 1 : SMALL PERTURBATOIN TEST
C       ISHFT = 2 : TOROIDAL SHIFT TEST
C
      READ (5,*) ISHFT
C
      NPSIM = NOSIM - 1
      DO 20 IOSIM = 1, NPSIM
C
      DO 25 LSTEP = 1, NSTEP
      SVALU(IOSIM,LSTEP) = 0.
   25 CONTINUE
C
      IF ( ISHFT.EQ.1 ) THEN
C
C     DEFINE PERTURBATION DISTANCE AND USE RANDOM DISTANCE
C
         IF ( IOSIM.EQ.1 ) DSEED = 123457.0
         CALL GGUBS ( DSEED, 2, RSCLE )
         XCOTR = RSCLE(1) * TSTEP * 2. - TSTEP / 2.
         YCOTR = RSCLE(2) * TSTEP * 2. - TSTEP / 2.
         DO 30 IPLUS = NPLU1, NPLU2
         JPLUS = IPLUS - NOFPT
         CPSND(JPLUS,1) = CPFST(IPLUS,1) + XCOTR
         CPSND(JPLUS,2) = CPFST(IPLUS,2) + YCOTR
  30     CONTINUE
      ELSE
         IF (IOSIM.EQ.1) DSEED = 123457.0
         CALL GGUBS( DSEED, 2, RSCLE )
         XCOTR = RSCLE(1) * XRANG
         YCOTR = RSCLE(2) * YRANG
         DO 40 IPLUS = NPLU1, NPLU2
         JPLUS = IPLUS - NOFPT
         CPSND(JPLUS,1) = CPFST(IPLUS,1) + XCOTR
         CPSND(JPLUS,2) = CPFST(IPLUS,2) + YCOTR
         IF (CPSND(JPLUS,1).LT.XBOT ) CPSND(JPLUS,1)=CPSND(JPLUS,1)+XRANG
         IF (CPSND(JPLUS,1).GT.XRANG) CPSND(JPLUS,1)=CPSND(JPLUS,1)-XRANG
         IF (CPSND(JPLUS,2).LT.YBOT ) CPSND(JPLUS,2)=CPSND(JPLUS,2)+YRANG
         IF (CPSND(JPLUS,2).GT.YRANG) CPSND(JPLUS,2)=CPSND(JPLUS,2)-YRANG
   40    CONTINUE
         GO TO 55
      ENDIF
C
C     ADJUST THE TRANSFORMED COORDINATE
C
      DO 50 IOPNT = 1, NOLPT
      IF (CPSND(IOPNT,1).LT.XBOT ) CPSND(IOPNT,1)=XBOT
      IF (CPSND(IOPNT,1).GT.XRANG) CPSND(IOPNT,1)=XRANG
      IF (CPSND(IOPNT,2).LT.YBOT ) CPSND(IOPNT,2)=YBOT
      IF (CPSND(IOPNT,2).GT.YRANG) CPSND(IOPNT,2)=YRANG
   50 CONTINUE
C
C     CALCULATE THE 2nd MOMENT
C
   55 CONTINUE
      DO 60 IOPNT = 1, NOFPT
      XCOOR = CPFST(IOPNT,1)
      YCOOR = CPFST(IOPNT,2)
      CALL BVSMM ( XCOOR, YCOOR, CPSND, SVALU, IOSIM )
   60 CONTINUE
   20 CONTINUE
C
C     MAPPED PATTERN CASE
C
      IOSIM = NOSIM
      DO 80 JOPNT = NPLU1,NPLU2
      KOPNT = JOPNT - NOFPT
      CPSND(KOPNT,1) = CPFST(JOPNT,1)
      CPSND(KOPNT,2) = CPFST(JOPNT,2)
   80 CONTINUE
C
      DO 75 LSTEP = 1, NSTEP
      SVALU(NOSIM,LSTEP) = 0.
   75 CONTINUE
      DO 70 IOPNT = 1, NOFPT
      XCOOR = CPFST(IOPNT,1)
      YCOOR = CPFST(IOPNT,2)
      CALL BVSMM ( XCOOR, YCOOR, CPSND, SVALU, IOSIM )
   70 CONTINUE
C
C     CUMULATIVE K FUNCTION
C
      DO 300 IOSIM = 1, NOSIM
      DO 200 ISTEP = 1, NSTEP
      ADDTV = 0.
      IF ( ISTEP.EQ.1 ) GO TO 200
      KSTEP = ISTEP - 1
      ADDTV = SVALU(IOSIM,KSTEP)
      SVALU(IOSIM,ISTEP) = SVALU(IOSIM,ISTEP) + ADDTV
  200 CONTINUE
  300 CONTINUE
C
      CALL KSTAT ( SVALU )
C
C     MONTE - CARLO STATISTICS
C
      WRITE(6,900)
      DO 110 IOSIM = 1, NOSIM
      CSTAT = 0.
      RSTAT = 0.
      DO 100 ISTEP = 1, NSTEP
      JSTAT = 0
      XSTAT = 0.
      ZSTAT = SVALU(IOSIM,ISTEP)
      DO 90 JOSIM = 1, NOSIM
      IF ( JOSIM.EQ.IOSIM ) GO TO 90
      XSTAT = XSTAT + SVALU(JOSIM,ISTEP)
   90 CONTINUE
      XSTAT = XSTAT / ( NOSIM - 1 )
      CSTAT = ( ZSTAT - XSTAT )**2
      RSTAT = RSTAT + CSTAT
  100 CONTINUE
      IF ( IOSIM.EQ.1 ) THEN
           RMINV = RSTAT
           RMAXV = RSTAT
      ENDIF
      IF ( IOSIM.NE.NOSIM.AND.RSTAT.LE.RMINV ) RMINV = RSTAT
      IF ( IOSIM.NE.NOSIM.AND.RSTAT.GE.RMAXV ) RMAXV = RSTAT
      WRITE(6,910) IOSIM, RSTAT
  110 CONTINUE
      WRITE(6,920) RMINV, RMAXV, RSTAT
C
  900 FORMAT(//,5X,'MONTE - CARLO TEST',//)
  910 FORMAT(   5X,'STATISTIC',I5,'    = ',F15.3 )
  920 FORMAT(//,5X,'MIN. VALUE OF STATISTIC    = ',F15.3,
     *         /,5X,'MAX. VALUE OF STATISTIC    = ',F15.3,
     *         /,5X,'MAPPED VALUE               = ',F15.3 )
C
```

```
      RETURN
      END
CC
C
      SUBROUTINE BVSMM ( XCOOR, YCOOR, CPSND, SVALU, IOSIM )
C
C     SUBROUTINE BVSMM EVALUATE THE SECOND MOMENT MEASURE FOR
C     BIVARIATE POINT PROCESS
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *              NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *              MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION CPSND(138,2), SVALU(NOSIM,NSTEP)
C
      CNSTV = 1.
C
C     CALCULATE THE WEIGHTING COEFFICIENT W(X,U)
C
      DIST1 = AMIN1 ( XCOOR, (XRANG - XCOOR) )
      DIST2 = AMIN1 ( YCOOR, (YRANG - YCOOR) )
      SQDIS = DIST1**2 + DIST2**2
      DO 20 IPOIN = 1, NOLPT
      XCOR1 = CPSND(IPOIN,1)
      YCOR1 = CPSND(IPOIN,2)
      DIST0 = SQRT((XCOOR-XCOR1)**2+(YCOOR-YCOR1)**2 )
C
C     NUMERICAL ERROR TERM
C
      WEIGT = 0.
      IF ( DIST0.LE.0.1 ) THEN
           WEIGT = 1.
           GO TO 15
      ENDIF
C
      ISTEP = DIST0 / DSTEP + 1
      IF ( ISTEP.GT.NSTEP ) GO TO 20
      IF ( DIST0**2.LE.SQDIS ) THEN
           DIST3 = ACOS( CNSTV * AMIN1(DIST1,DIST0) / DIST0 )
           DIST4 = ACOS( CNSTV * AMIN1(DIST2,DIST0) / DIST0 )
           WEIGT = 1. - ( DIST3 + DIST4 ) / 3.14159
      ELSE
           DIST3 = ACOS( CNSTV * DIST1 / DIST0 )
           DIST4 = ACOS( CNSTV * DIST2 / DIST0 )
           WEIGT = 0.75 - ( DIST3 + DIST4 ) / 6.28318
      ENDIF
   15 CONTINUE
      WEIGT = XRANG * YRANG / ( WEIGT * (NOLPT*NOFPT) )
C
C     CALCULATE K FUNCTION
C
      SVALU(IOSIM,ISTEP) = SVALU(IOSIM,ISTEP) + WEIGT
   20 CONTINUE
C
      RETURN
      END
CC
C
      SUBROUTINE NNMLE ( COORD, TRACE, SECON, XLAMO )
C
C     SUBROUTINE NNMLE EVALUATES THE HIERARCHICAL FIBER MODEL
C     WITH NEAREST NEIGHBOR FIBERS.
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION COORD(NOPNT,4), TRACE(NOFPT), SECON(NOPNT)
      DIMENSION COORW(138,2)
C
      TWOPI = 6.28318
      JITRT = 1
      VALTC = 0.
      NPLU1 = NOFPT + 1
      NPLU2 = NOFPT + NOLPT
      WRITE(6,910)
C
C     CALCULATE THE MLE OF THE NEAREST NEIGHBOR FIBER FUNCTION
C
      DO 10 IOPNT = NPLU1,NPLU2
C
C     READ MID POINT OF THE LATTER SET ( MAPPED DATA )
C
      READ (5,*) ( COORD(IOPNT,IDOFN), IDOFN=1,NDOF2)
      XCOOR = ( COORD(IOPNT,1) + COORD(IOPNT,3) ) / 2.
      YCOOR = ( COORD(IOPNT,2) + COORD(IOPNT,4) ) / 2.
      JOPNT = IOPNT - NOFPT
      COORW(JOPNT,1) = XCOOR
      COORW(JOPNT,2) = YCOOR
   10 CONTINUE
C
      JITRT = 1
C
C     EVALUATE LOG SUM OF MLE
C
      SIGMO = SIGMA
      DO 40 JSTEP = 1, NSTEP
      XLAMO = XLAMO + DSTEP
      DO 30 ISTEP = 1, NSTEP
      VALUE = 0.
      IF ( ISTEP.EQ.1 ) SIGMA = SIGMO
      SIGMA = SIGMA + DSTEP
C
      DO 20 IOPNT = 1, NOLPT
C
C     FIND THE SHORTEST DISTANCE BETWEEN A POINT AND A FIBER
C
      XCOOR = COORW(IOPNT,1)
      YCOOR = COORW(IOPNT,2)
      CALL NEARF  ( JITRT,XCOOR,YCOOR,FDIST,COORD,TRACE,SECON )
      JITRT = 2
C
C     EVALUATE MLE FOR NEAREST NEIGHBOR DISTANCE USING
C     MIXED DISTRIBUTION
C
      VALUE = VALUE + ALOG( XLAMO / (22.*14.) +
     *        (1.-XLAMO) * EXP( -0.5 * (FDIST/SIGMA)**2 ) /
     *         ( SIGMA * SQRT(TWOPI) * FLOAT(NOLPT)   ) )
   20 CONTINUE
      WRITE(6,920) SIGMA, XLAMO, VALUE
      IF ( ISTEP.EQ.1.AND.JSTEP.EQ.1 ) THEN
         VALMX = VALUE
         SIGMX = SIGMA
         XLAMX = XLAMO
      ENDIF
      IF ( VALUE.GT.VALMX ) THEN
         VALMX = VALUE
         SIGMX = SIGMA
         XLAMX = XLAMO
      ENDIF
   30 CONTINUE
   40 CONTINUE
      WRITE(6,930)
      WRITE(6,920) SIGMX, XLAMX, VALMX
C
  910 FORMAT(//,5X,'MLE OF NEAREST NEIGHBOR FIBER FUNCTION',//)
  920 FORMAT( /,2X,'SIGMA , XLAMO AND LOG SUM OF ML VALUE = ',
     *          2F10.5,5X,F15.7)
  930 FORMAT(//,5X,'MAX. VALUES',//)
C
      RETURN
      END
CC
C
      SUBROUTINE KFMLE  ( COORD,CORFB,SECON,TRACE,CORLF,COORL,SVALU,
     *                    XLAMO )
C
C     SUBROUTINE KFMLE EVALUATES THE HIERARCHICAL FIBER MODEL
C     WITH LINE - KERNEL FUNCTION.
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION COORD(NOPNT,4), CORFB(NOFPT,4), SECON(NOPNT),
     *          CORLF(NOLPT,4), COORL(NOLPT,2), TRACE(NOFPT)
      DIMENSION SVALU(NOSIM,NSTEP), RSCLE(2)
      DIMENSION COORW(138,2)
C
C        CALCULATE THE SHAPE OF THE LINE - KERNEL FUNCTION WITH
C        MAPPED DATA.
C
      TWOPI = 6.28318
      JITRT = 1
      VALTC = 0.
      NPLU1 = NOFPT + 1
      NPLU2 = NOFPT + NOLPT
      WRITE(6,910)
C
C
C     CALCULATE THE MLE OF THE LINE - KERNEL FUNCTION
C
      DO 10 IOPNT = NPLU1,NPLU2
C
C     READ MID POINT OF THE LATTER SET ( MAPPED DATA )
C
      READ (5,*) ( COORD(IOPNT,IDOFN), IDOFN=1,NDOF2)
      XCOOR = ( COORD(IOPNT,1) + COORD(IOPNT,3) ) / 2.
      YCOOR = ( COORD(IOPNT,2) + COORD(IOPNT,4) ) / 2.
      JOPNT = IOPNT - NOFPT
      COORW(JOPNT,1) = XCOOR
      COORW(JOPNT,2) = YCOOR
   10 CONTINUE
C
C     CALCULATE THE SUM OF ML
C
      SIGMO = SIGMA
      DO 40 JSTEP = 1, NSTEP
      XLAMO = XLAMO + DSTEP
      DO 20 ISTEP = 1, NSTEP
      VALTC = 0.
      VALUE = 0.
      IF ( ISTEP.EQ.1 ) SIGMA = SIGMO
      SIGMA = SIGMA + DSTEP
      DO 30 IOPNT = 1, NOLPT
      XCOOR = COORW(IOPNT,1)
      YCOOR = COORW(IOPNT,2)
      CALL LKERN ( JITRT,XCOOR,YCOOR,TRACE,VALU1,COORD,SECON,
     *             NTOFB   )
      JITRT = 2
      VALUE = VALUE + ALOG( XLAMO / ( 22. * 14. ) +
     *         (1.-XLAMO) * VALU1 / FLOAT(NOLPT) )
   30 CONTINUE
      WRITE(6,920) SIGMA, XLAMO, VALUE
      IF ( ISTEP.EQ.1.AND.JSTEP.EQ.1 ) THEN
           VALMX = VALUE
           SIGMX = SIGMA
           XLAMX = XLAMO
      ENDIF
      IF ( VALUE.GT.VALMX ) THEN
           VALMX = VALUE
           SIGMX = SIGMA
           XLAMX = XLAMO
      ENDIF
   20 CONTINUE
   40 CONTINUE
C
      WRITE(6,930)
      WRITE(6,920) SIGMX, XLAMX, VALMX
```

```
C
  910 FORMAT(//,5X,'MLE OF LINE - KERNEL FUNCTION',
     *        / )
  920 FORMAT( /,2X,'SIGMA AND XLAMO & LOG SUM OF ML VALUE  =',
     *        2F10.5,5X,F15.7)
  930 FORMAT(//,'MAX. VALUES',//)
C
      RETURN
      END
CC
C
      SUBROUTINE KERNL ( NOLPT, XCOR2, YOCR2, VLUKL, COORW, SIGMA )
C
C     SUBROUTINE KERNL EVALUATE THE BIVARIATE KERNEL FUNCTION WHEN
C     GENERATING THE 2nd SET.
C
      DIMENSION COORW(138,2)
C
      TWOPI = 6.28318
      VLUKL = 0.
C
      DO 10 IOPNT = 1, NOLPT
      XCORD = COORW(IOPNT,1)
      YCORD = COORW(IOPNT,2)
      DISTS = ( XCOR2-XCORD )**2 + ( YCOR2-YCORD )**2
      VLUKL = VLUKL + EXP( -0.5 * DISTS / SIGMA**2 )
   10 CONTINUE
      VLUKL = VLUKL / FLOAT(NOLPT)
C
      RETURN
      END
CC
C
      SUBROUTINE NEARF ( JITRT,XCOOR,YCOOR,FDIST,COORD,TRACE,
     *                   SECON )
C
C     SUBROUTINE NEARF CALCULATES THE NEAREST NEIGHBOR FIBER
C     DISTANCE FROM A GIVEN POINT IN A MAP
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION COORD(NOPNT,4), TRACE(NOFPT), SECON(NOPNT)
C
      FDIST = 0.
C
      IF ( JITRT.EQ.1 ) THEN
         DO 10 IOPNT = 1, NOFPT
         IF ( COORD(IOPNT,1).GT.COORD(IOPNT,3) ) THEN
                CORXP = COORD(IOPNT,3)
                CORYP = COORD(IOPNT,4)
                COORD(IOPNT,3) = COORD(IOPNT,1)
                COORD(IOPNT,4) = COORD(IOPNT,2)
                COORD(IOPNT,1) = CORXP
                COORD(IOPNT,2) = CORYP
         ENDIF
C
C     CALCULATE THE LINE EQUATION FOR THE GIVEN FIBER
C
C         y = secod * ( x - Xi ) + Yi
C         trace = trace length
C
         SECOD = ( COORD(IOPNT,4) - COORD(IOPNT,2) ) /
     *         ( COORD(IOPNT,3) - COORD(IOPNT,1) )
         SECON(IOPNT) = SECOD
         TRACE(IOPNT) = SQRT( (COORD(IOPNT,3) - COORD(IOPNT,1))**2 +
     *                        (COORD(IOPNT,4) - COORD(IOPNT,2))**2 )
   10    CONTINUE
      ENDIF
C
C     LOOKING FOR A NEAREST NEIGHBOR FIBER FROM A GIVEN POINT
C
      DO 20 IOPNT = 1, NOFPT
      IF ( SECON(IOPNT).LT.0.001 ) SECON(IOPNT) = 0.001
      SECOD = SECON(IOPNT)
      XCROS = ( SECOD*COORD(IOPNT,1) + XCOOR/SECOD + YCOOR
     *          - COORD(IOPNT,2) ) / ( SECOD + 1./SECOD )
      IF ( XCROS.LE.COORD(IOPNT,1) ) XCROS = COORD(IOPNT,1)
      IF ( XCROS.GE.COORD(IOPNT,3) ) XCROS = COORD(IOPNT,3)
      YCROS = ( XCROS-COORD(IOPNT,1) ) * SECOD + COORD(IOPNT,2)
      FDIS1 = SQRT( (XCOOR-XCROS)**2 + (YCOOR-YCROS)**2 )
C
C     FIND THE SHORTEST DISTANCE
C
      IF ( IOPNT.EQ.1 ) FDIST = FDIS1
      IF ( FDIS1.LT.FDIST ) FDIST = FDIS1
   20 CONTINUE
C
      RETURN
      END
CC
C
      SUBROUTINE ORINT ( COORD, TRACE, SECON )
C
C     SUBROUTINE ORINT MEASURE THE ORIENTATION
C     CORRELATIONS BETWEEN SET 1 AND 2.
C
C     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONFL/NOPNT,NOFPT,NOLPT,XRANG,YRANG, XBOT, YBOT,NOSIM,
     *             NTOTL,NDOF2,NSTEP,DSTEP,NDOFN,SIGMA,TSTEP,MAXXW,
     *             MAXYW,IOPTN,JOPTN,LOPTN,DFACT
      DIMENSION COORD(NOPNT,4), TRACE(NOFPT), SECON(NOPNT)
      DIMENSION CRORN(40,18), IRINT(20)
C
      TWOPI = 6.28318
      NPLU1 = NOFPT + 1
C
      IF ( IOPTN.EQ.4 ) GO TO 110
C
```

```
      READ (5,*) ANGLE
      NOANG = 180. / ANGLE
C
C     READ COORDINATES OF FIBERS
C
      DO 10 IOPNT = 1, NOPNT
      READ (5,*) ( COORD(IOPNT,JOPNT), JOPNT=1,NDOF2 )
   10 CONTINUE
C
C     LOOKING FOR A NEAREST NEIGHBOR FIBER
C
      JITRT = 1
      DO 20 IOPNT = NPLU1, NOPNT
      XCOOR = ( COORD(IOPNT,1) + COORD(IOPNT,3) ) / 2.
      YCOOR = ( COORD(IOPNT,2) + COORD(IOPNT,4) ) / 2.
      CALL NEARF ( JITRT,XCOOR,YCOOR,FDIST,COORD,TRACE,SECON )
      JITRT = 2
C
      IDIST = FDIST / DSTEP
      IDIST = IDIST + 1
      CORDD =  COORD(IOPNT,3) - COORD(IOPNT,1)
      IF ( CORDD.LE. 0.01.AND.CORDD.GE.0. ) CORDD = 0.01
      IF ( CORDD.GE.-0.01.AND.CORDD.LE.0. ) CORDD =-0.01
      STRKE = ( COORD(IOPNT,4) - COORD(IOPNT,2) ) / CORDD
      STRKE = ATAN(STRKE) * 180. / 3.14159
      IF ( STRKE.LT.0. ) STRKE = STRKE + 180.
      IORNT = STRKE / ANGLE
      IORNT = IORNT + 1
      CRORN(IDIST,IORNT) = CRORN(IDIST,IORNT) + 1
   20 CONTINUE
C
      WRITE(6,910)
      DO 30 ISTEP = 1, NSTEP
      DISTS = ISTEP * DSTEP
      DO 40 JSTEP = 1, NOANG
      STRKE = JSTEP * ANGLE
      IFREQ = CRORN(ISTEP,JSTEP)
      FREQY = FLOAT( IFREQ ) / FLOAT( NOLPT )
      WRITE(6,920) DISTS, STRKE, IFREQ, FREQY
   40 CONTINUE
   30 CONTINUE
C
      WRITE(6,930)
      DO 50 JSTEP = 1, NOANG
      STRKE = JSTEP * ANGLE
      DO 60 ISTEP = 1, NSTEP
      IRINT(ISTEP) = CRORN(ISTEP,JSTEP)
   60 CONTINUE
      WRITE(6,940) STRKE, (IRINT(ISTEP),ISTEP=1,NSTEP)
   50 CONTINUE
      RETURN
C
C     IOPTN = 4 : MLE OF ORIENTATION DISTRIBUTION
C
C     WRAPPED NORMAL DISTRIBUTION IS ADOPTED IN  CURRENT VERSION
```

```
C
  110 CONTINUE
      DO 115 IOPNT = 1, NOFPT
      READ(5,*) DUMM1, DUMM2, DUMM3, DUMM4
  115 CONTINUE
      DO 120 IOPNT = 1, NOLPT
      READ (5,*) ( COORD(IOPNT,JOPNT),JOPNT=1,NDOF2 )
      DIFFS = COORD(IOPNT,3) - COORD(IOPNT,1)
      IF ( DIFFS.LE. 0.01.AND.DIFFS.GE.0. ) DIFFS = 0.01
      IF ( DIFFS.GE.-0.01.AND.DIFFS.LE.0. ) DIFFS =-0.01
      SECOD = ( COORD(IOPNT,4) - COORD(IOPNT,2) ) / DIFFS
      SECON(IOPNT) = ATAN( SECOD ) * 57.2958
      IF ( SECON(IOPNT).LT.0. ) SECON(IOPNT) = SECON(IOPNT) + 180.
  120 CONTINUE
C
C     READ OPTION ( MOPTN )
C       1 : Von Mises DISTRIBUTION ON A CIRCLE
C       2 : WRAPPED NORMAL DISTRIBUTION
C
      NOLP2 = NOLPT * 2
      READ (5,*) MOPTN
      WRITE(6,990) MOPTN
      IF ( MOPTN.EQ.2 ) GO TO 200
C
C     MOPTN = 1
C     CALCULATE THE MLE OF GIVEN DATA USING  Von Mises DISTRIBUTION
C
      XKVAL = TSTEP
      XANGL = SIGMA
      DO 150 JSTEP = 1, NSTEP
      XANG1 = XANGL * JSTEP
      XANG2 = XANG1 + 180.
      DO 140 ISTEP = 1, NSTEP
      XKVAL = TSTEP + ( ISTEP - 1 ) * DSTEP
C
      BESSL = 1. + (0.5*XKVAL)**2 + (0.5*XKVAL)**4 / 4. +
     *        (0.5*XKVAL)**6 / 36. + (0.5*XKVAL)**8 / (24.)**2 +
     *        (0.5*XKVAL)**10 / (120.)**2
      BESS1 = 1. / ( TWOPI * BESSL )
C
      VALUE = 0.
      DO 130 IOPNT = 1, NOLP2
      IF ( IOPNT.LE.NOLPT ) THEN
           ANGLE = SECON(IOPNT)
      ELSE
           JOPNT = IOPNT - NOLPT
           ANGLE = SECON(JOPNT) + 180.
      ENDIF
C
      VALUE = VALUE + ALOG( 0.5 * BESS1 * (   EXP( XKVAL *
     *        COS( (ANGLE-XANG1)*0.01745 ) ) ) + 0.5 * BESS1
     *   * ( EXP(XKVAL * COS( (ANGLE-XANG2)*0.01745 ) ) ) )
  130 CONTINUE
C
      WRITE(6,950) XKVAL, XANG1, VALUE
      IF ( ISTEP.EQ.1.AND.JSTEP.EQ.1 ) THEN
        XKMAX = XKVAL
        XAMAX = XANG1
        VALMX = VALUE
      ENDIF
      IF ( VALUE.GT.VALMX ) THEN
        XKMAX = XKVAL
        XAMAX = XANG1
        VALMX = VALUE
      ENDIF
  140 CONTINUE
  150 CONTINUE
      GO TO 300
C
C     MOPTN = 2
C     WRAPPED NORMAL DISTRIBUTION
C
  200 CONTINUE
      XANGL = SIGMA
      KSTEP = -NSTEP / 2
      DO 230 ISTEP = 1, NSTEP
      XANG1 = XANGL + ( ISTEP-1) * 5.
      XANG2 = XANG1 + 180.
C
      DO 220 JSTEP = 1, NSTEP
      VALUE = 0.
      XKVAL = TSTEP + ( JSTEP - 1 ) * DSTEP
      DO 210 IOPNT = 1, NOLP2
      IF ( IOPNT.LE.NOLPT ) THEN
           ANGLE = SECON(IOPNT)
      ELSE
           JOPNT = IOPNT - NOLPT
           ANGLE = SECON(JOPNT) + 180.
      ENDIF
      VALU1 = 0.
      VALU2 = 0.
C
      DO 240 LSTEP = 1, NSTEP
        RSTEP = KSTEP + LSTEP
        VALU1 = VALU1 + EXP( -0.5 * ( ( (ANGLE-XANG1-360.*RSTEP)
     *      * 0.01745  ) **2 / XKVAL**2 ) )
        VALU2 = VALU2 + EXP( -0.5 * ( ( (ANGLE-XANG2-360.*RSTEP)
     *      * 0.01745  ) **2 / XKVAL**2 ) )
  240   CONTINUE
      VALUE = VALUE +
     *        ALOG( 0.5 / (XKVAL*SQRT(TWOPI)) * (VALU1+VALU2))
C
  210 CONTINUE
      WRITE(6,950) XKVAL, XANG1, VALUE
      IF ( ISTEP.EQ.1.AND.JSTEP.EQ.1 ) THEN
           XKMAX = XKVAL
           XAMAX = XANG1
           VALMX = VALUE
      ENDIF
      IF ( VALUE.GT.VALMX ) THEN
```

```
          XKMAX = XKVAL
          XAMAX = XANG1
          VALMX = VALUE
      ENDIF
  220 CONTINUE
  230 CONTINUE
C
  300 CONTINUE
C
      WRITE(6,960)
      WRITE(6,950) XKMAX, XAMAX, VALMX
C
C     CALCULATE FREQUENCIES OF THE WRAPPED NORMAL DISTRIBUTION
C     WITH CALCUALTED MAX. VALUE OF SIGMA
C
      WRITE(6,970)
      VALUE = 0.
      VALMX = 0.
      SIGMA = XKMAX
      XKVAL = XKMAX
      XANG1 = XAMAX
      XANG2 = XANG1 + 180.
      DO 330 IANGL = 1, 36
      ANGLE = IANGL * 10.
      VALUE = 0.
      VALU1 = 0.
      VALU2 = 0.
      IF ( MOPTN.EQ.2 ) THEN
         DO 320 JSTEP = 1, NSTEP
         RSTEP = KSTEP + JSTEP
         VALU1 = VALU1 + EXP( -0.5 * ( ( (ANGLE-XANG1-360.*RSTEP)
     *          * 0.01745  )**2 / SIGMA**2 ) )
         VALU2 = VALU2 + EXP( -0.5 * ( ( (ANGLE-XANG2-360.*RSTEP)
     *          * 0.01745  )**2 / SIGMA**2 ) )
  320    CONTINUE
         VALUE = 0.5 / (SIGMA*SQRT(TWOPI)) * (VALU1+VALU2)
         VALMX = VALMX + VALUE
      ELSE
         BESSL = 1. + (0.5*XKVAL)**2 + (0.5*XKVAL)**4 / 4. +
     *        (0.5*XKVAL)**6 / 36. + (0.5*XKVAL)**8 / (24.)**2 +
     *        (0.5*XKVAL)**10 / (120.)**2
         BESS1 = 1. / ( TWOPI * BESSL )
C
         VALUE =  BESS1 * ( 0.5 *  EXP( XKVAL *
     *            COS( (ANGLE-XANG1)*0.01745 ) ) + 0.5 *
     *        EXP( XKVAL * COS( (ANGLE-XANG2)*0.01745 ) ) )
         VALMX = VALMX + VALUE
      ENDIF
      WRITE(6,980) ANGLE, VALUE
  330 CONTINUE
      VALMX = VALMX * 0.17453
      WRITE(6,890) MOPTN,VALMX
C
  890 FORMAT(//,5X,'OPTION = ',I5,
     *        /,5X,'INTEGRATION OF FREQUENCY = ',F15.7)
  910 FORMAT(//,5X,'ORIENTATION CORRELATION MEASURE',
     *        /,7X,'DISTANCE   ANGLE   FREQUENCY',//)
  920 FORMAT(   5X,2(F7.3,3X),I3,3X,F9.6)
  930 FORMAT(//,5X,'ANGLE DISTANCE : ( dstep * integer ) m',//)
  940 FORMAT(   3X,F7.3,3X,10I5)
  950 FORMAT(   5X,'k-value ANGLE & LOG SUM = ',3F15.6)
  960 FORMAT(//,5X,' MAX. VALUES OF WRAPPED NORMAL DISTRIBUTION'//)
  970 FORMAT(//,5X,'FREQUENCY EVALUATION WITH GIVEN MAX. VALUES',//)
  980 FORMAT(   5X,'ANGLE & FREQUENCY = ',2F15.5)
  990 FORMAT(//,5X,'MLE OPTION : MOPTN ',10X,I5,
     *        /,8X,'1 : Von Mises DISTRIBUTION ON A CIRCLE',
     *        /,8X,'2 : WRAPPED NORMAL DISTRIBUTION       ',
     *       //)
      RETURN
      END
      PROGRAM TDIST
C
C     PROGRAM TDIST ESTIMATE THE TRACE LENGTH DISTRIBUTION
C     USING MAXIMUM LIKELIHOOD ESTIMATE
C
C     THREE POSSIBLE DISTRIBUTIONS ARE EXAMINED
C         1.  EXPONENTIAL DISTRIBUTION
C         2.  TRUNCATED NORMAL DISTRIBUTION
C         3.  LOGNORMAL DISTRIBUTION
C
C     TRACE(i,1)  : TRACE LENGTH DATA
C     TRACE(i,2)  : BOUNDARY OF THE MAPPED AREA
C     NCONF(i)    : CHARACTERISTICS OF THE TRACE
C                   1 FOR THE TRACE WITH BOTH ENDS OBSERVABLE
C                   2 FOR THE TRACE WITH ONE END OBSERVABLE
C                   3 FOR THE TRACE WITH NO END OBSERVABLE
C
      DIMENSION TRACE(39,2), NCONF(39)
C
      OPEN (5, FILE='tdist.dat5', STATUS='UNKNOWN')
      OPEN (6, FILE='tdist.out')
C
      NOFPT = 39
C
      DO 10 IOFPT = 1,NOFPT
      READ (5,*) TRACE(IOFPT,1), TRACE(IOFPT,2), NCONF(IOFPT)
      TRACE(IOFPT,1) = 0.833 * TRACE(IOFPT,1)
      TRACE(IOFPT,2) = 0.833 * TRACE(IOFPT,2)
   10 CONTINUE
C
C     ITERATE FOR EACH CASE
C
      SUMMX = 0.
      SUMMS = 0.
      READ (5,*) NITER
      DO 20 IITER = 1,NITER
C
C     READ THE OPTION
C
C         NOPTN = 1 : EXPONENTIAL DISTRIBUTION WITH MEAN VALUE
```

```
C                 2 : TRUNCATED NORMAL DISTRIBUTION WITH TWO VARIABLES
C                 3 : LOGNORMAL DISTRIBUTION WITH TWO VARIABLES
C
      READ (5,*) NOPTN
      GO TO ( 100, 200, 300 ), NOPTN
C
C     EXPONENTIAL DISTIBUTION WITH ASSUMED MEAN VALUE
C
  100 CONTINUE
      WRITE(6,900)
      READ (5,*) RMEAN, STEPS, NSTEP
      DO 40 ISTEP = 1,NSTEP
      PROB3 = 0.
      PROB6 = 0.
      PROB9 = 0.
      XMEAN = RMEAN + (ISTEP - 1 ) * STEPS
      DO 30 IOFPT = 1, NOFPT
      JOPTN = NCONF(IOFPT)
      RTRCE = TRACE(IOFPT,1)
      BTRCE = TRACE(IOFPT,2)
      CONST = 1. / ( 3.*XMEAN + BTRCE - 3.*XMEAN*EXP(-BTRCE/XMEAN) )
C      CONST = 1. / ( XMEAN + BTRCE - XMEAN * EXP( -BTRCE/XMEAN ) )
      IF ( JOPTN.EQ.1 ) THEN
           PROB1 = BTRCE - XMEAN + XMEAN *  EXP( - BTRCE / XMEAN )
           PROB1 = PROB1 * CONST
           PROB2 = ( BTRCE - RTRCE ) * EXP( -RTRCE / XMEAN ) / XMEAN
           PROB3 = PROB3 + ALOG( PROB1 * PROB2 )
      ENDIF
      IF ( JOPTN.EQ.2 ) THEN
           PROB4 = 2.* XMEAN * ( 1. - EXP( - BTRCE / XMEAN ) )
           PROB4 = PROB4 * CONST
           PROB5 = EXP( - RTRCE / XMEAN ) / ( XMEAN * ( 1. -
     *             EXP( - BTRCE / XMEAN ) ) )
           PROB6 = PROB6 + ALOG( PROB4 * PROB5 )
      ENDIF
      IF ( JOPTN.EQ.3 ) THEN
           PROB7 = XMEAN * EXP( - BTRCE / XMEAN )
           PROB7 = PROB7 * CONST
           PROB8 = 1.
           PROB9 = PROB9 +ALOG( PROB7*PROB8 )
      ENDIF
      PROBQ = PROB3 + PROB6
   30 CONTINUE
C
C     FIND MAXIMUM LIKELIHOOD ESTIMATES
C
      WRITE(6,910) XMEAN, PROB3, PROB6, PROB9
      IF ( ISTEP.EQ.1 ) THEN
           SUMMA = PROBQ
           SUMM3 = PROB3
           SUMM6 = PROB6
           SUMM9 = PROB9
           VMEA3 = XMEAN
           VMEA6 = XMEAN
           VMEA9 = XMEAN
      ENDIF
      IF ( PROB3.GT.SUMM3 ) THEN
           SUMM3 = PROB3
           VMEA3 = XMEAN
      ENDIF
      IF ( PROB6.GT.SUMM6 ) THEN
           SUMM6 = PROB6
           VMEA6 = XMEAN
      ENDIF
      IF ( PROB9.GT.SUMM9 ) THEN
           SUMM9 = PROB9
           VMEA9 = XMEAN
      ENDIF
      IF ( PROBQ.GT.SUMMA ) THEN
           SUMMA = PROBQ
           VMEAQ = XMEAN
      ENDIF
   40 CONTINUE
      WRITE(6,920) VMEA3, SUMM3, VMEA6, SUMM6, VMEA9,SUMM9,VMEAQ,SUMMA
C
C     COMPARE THE RESULT WITH EPSTEIN'S FORMULA FOR SAMPLING LINE
C
      ISUMM = 0
      TLENG = 0.
      DO 50 IOFPT = 1,NOFPT
      KOPTN = NCONF(IOFPT)
      TLENG = TLENG + TRACE(IOFPT,1)
      IF ( KOPTN.EQ.1 ) ISUMM = ISUMM + 1
   50 CONTINUE
      VMEAN = TLENG / FLOAT( ISUMM )
      WRITE(6,930) VMEAN
C
      GO TO 400
C
C     TRUNCATED NORMAL DISTRIBUTION WITH ASSUMED MEAN AND STD. DEVIATION
C
  200 CONTINUE
      READ (5,*) XMEAN, DEVIA, STEP1, STEP2
      GO TO 400
C
C     LOGNORMAL DISTRIBUTION WITH ASSUMED MEAN STD. DEVIATION
C
  300 CONTINUE
      READ (5,*) XMEAN, DEVIA, STEP1, STEP2
C
  400 CONTINUE
   20 CONTINUE
C
  900 FORMAT(//,5X,'EXPONENTIAL DISTRIBUTIONS OF TRACES'
     *       /,8X,'MEAN VALUE',10X,'LOG SUM FOR TYPE 1    2    &    3',/)
  910 FORMAT(7X, F12.5,5X,3(F10.5,3X) )
  920 FORMAT(//,5X,'MAXIMUM LIKELIHOOD ESTIMATE FOR EXPONENTIAL CASE',
     *       //,5X,'FOR TYPE 1 ( BOTH ENDS VISIBLE )',
     *        /,7X,'MEAN VALUE = ',T20,F10.4,
     *        /,7X,'MAX. VALUE = ',T20,F10.4,
     *       //,5X,'FOR TYPE 2  ( ONE END VISIBLE )',
     *        /,7X,'MEAN VALUE = ',T20,F10.4,
     *        /,7X,'MAX. VALUE = ',T20,F10.4,
     *       //,5X,'FOR TYPE 3  ( NO END VISIBLE )',
     *        /,7X,'MEAN VALUE = ',T20,F10.4,
     *        /,7X,'MAX. VALUE = ',T20,F10.4,
     *       //,5X,'FOR SET 1  ( ALL TRACES)',
     *        /,7X,'MEAN VALUE = ',T20,F10.4,
     *        /,7X,'MAX. VALUE = ',T20,F10.4,//)
  930 FORMAT(//,5X,'MLE FOR MEAN VALUE BY EPSTEIN EQUATION =',F15.7)
C
      STOP
      END

      PROGRAM TERMN
C
C     PROGRAM TERMN EVALUATE THE TRACE LENGTH DISTRIBUTIONS
C     ACCORDING TO THE GIVEN SIMULATED TYPICAL POINTS.
C
      DIMENSION COORD(138,2), TDIST(138), RSCLE(1), TSEND(138),
     *          TRCPT(138,4), TFREQ(30),  RSCL1(1)
      DIMENSION CORFP(39,4),  TRACE(39),  SECON(39)
C
      NLENG = 30
C
C     FIRST, READ THE BEST FIT OF THE TYPICAL POINTS
C
      READ (5,*) NOPNT,NOFPT,NOSIM,VMEAN,AMEAN,SIGMA,TRUNL
      NOLPT = NOPNT - NOFPT
      READ (5,*)  XBOT,  XTOP,  YBOT,  YTOP
      WRITE(6,820) NOPNT,NOFPT,NOLPT,VMEAN, AMEAN, SIGMA
  820 FORMAT(//,5X,'NO. OF TOTAL POINTS     = ',I5,
     *        /,5X,'NO. OF POINTS IN SET 1 = ',I5,
     *        /,5X,'NO. OF POINTS IN SET 2 = ',I5,
     *        /,5X,'MEAN TRACE LENGTH OF SET 2     = ',F15.5,
     *        /,5X,'MEAN ANGLE OF SET 2            = ',F15.5,
     *        /,5X,'STANDARD DEVIATION OF ANGLE    = ',F15.5,//)
      XRANG = XTOP - XBOT
      YRANG = YTOP - YBOT
C
      READ (5,*) MOPTN
      WRITE(6,990) MOPTN
  990 FORMAT(//,5X,'ORIENTATION DISTRIBUTION OPTION = ',I10,
     *        /,8X,'( 1 : Von Mises DISTRIBUTION       )',
     *        /,8X,'( 2 : WRAPPED NORMAL DISTRIBUTION )',//)
      DO 10 IOPNT = 1, NOFPT
      READ (5,*) ( CORFP(IOPNT,JOPNT), JOPNT=1,4)
      CORRS = CORFP(IOPNT,3) - CORFP(IOPNT,1)
      IF ( CORRS.LT.0. ) THEN
         CORTX = CORFP(IOPNT,1)
         CORTY = CORFP(IOPNT,2)
         CORFP(IOPNT,1) = CORFP(IOPNT,3)
         CORFP(IOPNT,2) = CORFP(IOPNT,4)
         CORFP(IOPNT,3) = CORTX
         CORFP(IOPNT,4) = CORTY
      ENDIF
      TRACE(IOPNT) = SQRT(  (CORFP(IOPNT,3)-CORFP(IOPNT,1))**2 +
     *                      (CORFP(IOPNT,4)-CORFP(IOPNT,2))**2 )
      IF ( CORRS.LE. 0.01.AND.CORRS.GE.0. ) CORRS = 0.01
      IF ( CORRS.GE.-0.01.AND.CORRS.LT.0. ) CORRS =-0.01
      SECON(IOPNT) = ( CORFP(IOPNT,4)-CORFP(IOPNT,2) ) / CORRS
   10 CONTINUE
C
      DO 20 IOPNT = 1, NOLPT
      READ (5,*) IDUMM, COORD(IOPNT,1), COORD(IOPNT,2)
   20 CONTINUE
C
      TWOPI = 6.28318
      XANGL = 105.
      IF ( MOPTN.EQ.1 ) THEN
        XKVAL = 1.9
        BESSL = 1. + (0.5*XKVAL)**2 + (0.5*XKVAL)**4 / 4. +
     *        (0.5*XKVAL)**6 / 36. + (0.5*XKVAL)**8 / (24.)**2
     *      + (0.5*XKVAL)**10 / (120.)**2
        BESS1 = 1. / (TWOPI * BESSL )
        VALMX = BESS1 * ( 0.5 * EXP(XKVAL) + 0.5 * EXP( -XKVAL))
      ELSE
        XKVAL = 0.8
        VALU1 = 0.
        VALU2 = 0.
        NSTEP = 10
        KSTEP = -NSTEP / 2
        DO 500 JSTEP = 1, NSTEP
        RSTEP = KSTEP + JSTEP
        VALU1 = VALU1 + EXP(-0.5 * ((( -360.*RSTEP)*0.01745 )**2
     *          / XKVAL**2 ) )
        VALU2 = VALU2 + EXP(-0.5 * (((-180.-360.*DSTEP)*0.01745)**2
     *          / XKVAL**2 ) )
 500    CONTINUE
        VALMX = 0.5 * ( VALU1 + VALU2 ) / ( XKVAL * SQRT(TWOPI) )
      ENDIF
C
      JOPNT = 0
      DSEED = 12345.D0
   30 CONTINUE
      CALL GGEXN ( DSEED, VMEAN, 1, RSCLE )
      IF ( RSCLE(1).LT.TRUNL ) GO TO 30
C
   35 CONTINUE
      CALL GGUBS ( DSEED, 1, RSCL1 )
      ORINT = RSCL1(1) * 180.
      IF ( MOPTN.EQ.1 ) THEN
        VALUE = BESS1 * ( 0.5 *EXP( XKVAL *
     *          COS( (ORINT-XANGL)*0.01745 ) ) +
     *          0.5*EXP(XKVAL*COS((ORINT-XANGL-180.)*0.01745)))
      ELSE
        VALU1 = 0.
        VALU2 = 0.
        DO 510 ISTEP = 1, NSTEP
        RSTEP = KSTEP + ISTEP
        VALU1 = VALU1 + EXP( -0.5 * ((( ORINT-XANGL-360.*
     *        RSTEP) * 0.01745 )**2 / XKVAL**2 ) )
          VALU2 = VALU2 + EXP( -0.5 * ((( ORINT-XANGL-180.-360.*
     *        RSTEP) * 0.01745 )**2 / XKVAL**2) )
  510     CONTINUE
          VALUE = 0.5*(VALU1+VALU2) / ( XKVAL * SQRT(TWOPI) )
      ENDIF
      CALL GGUBS ( DSEED, 1, RSCL1 )
      IF ( RSCL1(1).GT.(VALUE/VALMX) ) GO TO 35
C
      JOPNT = JOPNT + 1
      TDIST(JOPNT) = RSCLE(1)
      TDSTH = TDIST(JOPNT) / 2.
      TDST1 = SQRT( 1. + (TAN(ORINT))**2 )
      TRCPT(JOPNT,1) = COORD(JOPNT,1) + TDSTH / TDST1
      TRCPT(JOPNT,2) = COORD(JOPNT,2) + TDSTH * TAN(ORINT) / TDST1
      IF ( TRCPT(JOPNT,1).GT.XRANG ) THEN
         TRCPT(JOPNT,1) = XTOP
         TRCPT(JOPNT,2) = COORD(JOPNT,2) +
     *         ( TRCPT(JOPNT,1) - COORD(JOPNT,1) ) * TAN(ORINT)
      ENDIF
      IF ( TRCPT(JOPNT,2).GT.YRANG ) THEN
         TRCPT(JOPNT,2) = YTOP
         TRCPT(JOPNT,1) = COORD(JOPNT,1) +
     *         ( TRCPT(JOPNT,2) - COORD(JOPNT,2) ) / TAN(ORINT)
      ENDIF
C
      IF ( TRCPT(JOPNT,1).LT.XBOT ) THEN
         TRCPT(JOPNT,1) = XBOT
         TRCPT(JOPNT,2) = COORD(JOPNT,2) -
     *         ( COORD(JOPNT,1) - XBOT ) * TAN(ORINT)
      ENDIF
      YCOOR = -1.19 * TRCPT(JOPNT,1) + 7.78
      IF ( TRCPT(JOPNT,2).LT.YCOOR ) THEN
         TRCPT(JOPNT,1) = ( 7.78 + TAN(ORINT ) * COORD(JOPNT,1)
     *         - COORD(JOPNT,2) ) / ( 1.19 + TAN(ORINT) )
         TRCPT(JOPNT,2) = -1.19 * TRCPT(JOPNT,1) + 7.78
      ENDIF
      IF ( TRCPT(JOPNT,2).LT.YBOT ) THEN
         TRCPT(JOPNT,2) = YBOT
         TRCPT(JOPNT,1) = COORD(JOPNT,1) -
     *         ( COORD(JOPNT,2) - YBOT ) / TAN(ORINT)
      ENDIF
C
      TRCPT(JOPNT,3) = COORD(JOPNT,1) - TDSTH / TDST1
      TRCPT(JOPNT,4) = COORD(JOPNT,2) - TDSTH * TAN(ORINT) / TDST1
      IF ( TRCPT(JOPNT,3).LT.XBOT ) THEN
         TRCPT(JOPNT,3) = XBOT
         TRCPT(JOPNT,4) = COORD(JOPNT,2) -
     *         ( COORD(JOPNT,1) - TRCPT(JOPNT,3) ) * TAN(ORINT)
      ENDIF
      IF ( TRCPT(JOPNT,3).GT.XRANG ) THEN
         TRCPT(JOPNT,3) = XTOP
         TRCPT(JOPNT,4) = COORD(JOPNT,2) +
     *         ( TRCPT(JOPNT,3) - COORD(JOPNT,1) ) * TAN(ORINT)
      ENDIF
      YCOOR = -1.19 * TRCPT(JOPNT,3) + 7.78
      IF ( TRCPT(JOPNT,4).LT.YCOOR ) THEN
         TRCPT(JOPNT,3) = ( 7.78 + TAN(ORINT) * COORD(JOPNT,1)
     *        - COORD(JOPNT,2) ) / ( 1.19 + TAN(ORINT) )
         TRCPT(JOPNT,4) = -1.19 * TRCPT(JOPNT,3) + 7.78
      ENDIF
      IF ( TRCPT(JOPNT,4).LT.YBOT ) THEN
         TRCPT(JOPNT,4) = YBOT
         TRCPT(JOPNT,3) = COORD(JOPNT,1) -
     *        ( COORD(JOPNT,2) - TRCPT(JOPNT,4) ) / TAN(ORINT)
      ENDIF
      IF ( TRCPT(JOPNT,4).GT.YRANG ) THEN
         TRCPT(JOPNT,4) = YTOP
         TRCPT(JOPNT,3) = COORD(JOPNT,1) +
     *        ( TRCPT(JOPNT,4) - COORD(JOPNT,2) ) / TAN(ORINT)
      ENDIF
C
      YCOOR = 0.698 * TRCPT(JOPNT,1) - 4.56
      IF ( TRCPT(JOPNT,2).LT.YCOOR ) THEN
         TRCPT(JOPNT,1) = ( -4.56 + TAN(ORINT) * COORD(JOPNT,1)
     *        - COORD(JOPNT,2) ) / ( -0.698 + TAN(ORINT) )
         TRCPT(JOPNT,2) = 0.698 * TRCPT(JOPNT,1) - 4.56
      ENDIF
      YCOOR = 0.698 * TRCPT(JOPNT,3) - 4.56
      IF ( TRCPT(JOPNT,4).LT.YCOOR ) THEN
         TRCPT(JOPNT,3) = ( -4.56 + TAN(ORINT) * COORD(JOPNT,1)
     *        - COORD(JOPNT,2) ) / ( -0.698 + TAN(ORINT) )
         TRCPT(JOPNT,4) = 0.698 * TRCPT(JOPNT,3) - 4.56
      ENDIF
      YCOOR = 0.698 * TRCPT(JOPNT,1) + 11.
      IF (TRCPT(JOPNT,2).GT.YCOOR ) THEN
         TRCPT(JOPNT,1) = ( 11. + TAN(ORINT) * COORD(JOPNT,1)
     *        - COORD(JOPNT,2) ) / ( -0.698 + TAN(ORINT) )
         TRCPT(JOPNT,2) = 0.698 * TRCPT(JOPNT,1) + 11.
      ENDIF
      YCOOR = 0.698 * TRCPT(JOPNT,3) + 11.
      IF ( TRCPT(JOPNT,4).GT. YCOOR ) THEN
         TRCPT(JOPNT,3) = ( 11. + TAN(ORINT) * COORD(JOPNT,1)
     *        - COORD(JOPNT,2) ) / ( -0.698 + TAN(ORINT) )
         TRCPT(JOPNT,4) = 0.698 * TRCPT(JOPNT,3) + 11.
      ENDIF
C
      IF ( TRCPT(JOPNT,3).LT.TRCPT(JOPNT,1) ) THEN
           TRCTX = TRCPT(JOPNT,1)
           TRCTY = TRCPT(JOPNT,2)
           TRCPT(JOPNT,1) = TRCPT(JOPNT,3)
           TRCPT(JOPNT,2) = TRCPT(JOPNT,4)
           TRCPT(JOPNT,3) = TRCTX
           TRCPT(JOPNT,4) = TRCTY
      ENDIF
C
      TDIST(JOPNT) = SQRT(  ( TRCPT(JOPNT,3) - TRCPT(JOPNT,1) )**2 +
     *                      ( TRCPT(JOPNT,4) - TRCPT(JOPNT,2) )**2 )
      IF ( TDIST(JOPNT).LT.TRUNL ) THEN
           JOPNT = JOPNT - 1
           GO TO 30
      ENDIF
C
      CORRS = TRCPT(JOPNT,3) - TRCPT(JOPNT,1)
      IF ( CORRS.LE. 0.01.AND.CORRS.GE.0. ) CORRS = 0.01
      IF ( CORRS.GE.-0.01.AND.CORRS.LT.0. ) CORRS =-0.01
      TSEND(JOPNT) = ( TRCPT(JOPNT,4)-TRCPT(JOPNT,2) ) / CORRS
C
C     CALCULATE TERMINATION POINTS
C       IF TRACE LENGTH IS SMALLER THAN MEAN LENGTH, TERMINATE AT
C       THE NEAREST NEIGHBOR FIBER, IF IT IS GREATER THAN THAT OF MEAN
C       TERMINATE AT THE SECOND NEAREST NEIGHBOR FIBER.
C
      XMEAN = ( TRCPT(JOPNT,3) + TRCPT(JOPNT,1) ) / 2.
      YMEAN = ( TRCPT(JOPNT,4) + TRCPT(JOPNT,2) ) / 2.
C
      IFLAG = 0
      JFLAG = 0
      XMAXL = ( TRCPT(JOPNT,3) - TRCPT(JOPNT,1) ) / 2.
      XMINL = ( TRCPT(JOPNT,1) - TRCPT(JOPNT,3) ) / 2.
      XMAXO = XMAXL
      XMINO = XMINL
C
      DO 100 IOPNT = 1, NOFPT
      XCROS=(SECON(IOPNT)*CORFP(IOPNT,1)-TSEND(JOPNT)*TRCPT(JOPNT,1)
     *        + TRCPT(JOPNT,2) - CORFP(IOPNT,2) ) /
     *        ( SECON(IOPNT) - TSEND(JOPNT) )
      IF ( XCROS.GT.TRCPT(JOPNT,3).OR.XCROS.LT.TRCPT(JOPNT,1) )
     *       GO TO 100
      YCROS = SECON(IOPNT) * ( XCORS - CORFP(IOPNT,1) ) +
     *        CORFP(IOPNT,2)
      IF ( TRCPT(JOPNT,4).GE.TRCPT(JOPNT,2) ) THEN
           IF ( YCROS.GT.TRCPT(JOPNT,4).OR.YCROS.LT.TRCPT(JOPNT,2) )
     *          GO TO 100
      ELSE
           IF ( YCROS.GT.TRCPT(JOPNT,2).OR.YCROS.LT.TRCPT(JOPNT,4) )
     *          GO TO 100
      ENDIF
      TRILH = SQRT((XMEAN-XCROS)**2 + (YMEAN-YCROS)**2 )
      IF ( TRILH.GT.TDSTH ) GO TO 100
C
      IF ( XCROS.GE.XMEAN ) IFLAG = IFLAG + 1
      IF ( XCROS.LT.XMEAN ) JFLAG = JFLAG + 1
      XRESI = XCROS - XMEAN
      IF ( IFLAG.EQ.1 ) XMAXL = XRESI
      IF ( JFLAG.EQ.1 ) XMINL = XRESI
      IF ( IFLAG.GT.1.AND.XRESI.GT.XMAXL ) XMAXL = XRESI
      IF ( JFLAG.GT.1.AND.XRESI.LT.XMINL ) XMINL = XRESI
  100 CONTINUE
C
C     CONSIDER SET 2 EFFECT
C
      IF ( JOPNT.GT.1 ) THEN
        KOPNT = JOPNT - 1
        DO 110 KPONT = 1, KOPNT
        XCROS=(TSEND(KPONT)*TRCPT(KPONT,1)-TSEND(JOPNT)*TRCPT(JOPNT,1)
     *       + TRCPT(JOPNT,2) - TRCPT(KPONT,2) ) /
     *        ( TSEND(KPONT) - TSEND(JOPNT) )
        IF ( XCROS.GT.TRCPT(JOPNT,3).OR.XCROS.LT.TRCPT(JOPNT,1) )
     *     GO TO 110
        YCROS = TSEND(KPONT) * ( XCROS - TRCPT(KPONT,1) ) +
     *          TRCPT(KPONT,2)
        IF ( TRCPT(JOPNT,4).GE.TRCPT(JOPNT,2) ) THEN
             IF ( YCROS.GT.TRCPT(JOPNT,4).OR.YCROS.LT.TRCPT(JOPNT,2) )
     *            GO TO 110
        ELSE
             IF ( YCROS.GT.TRCPT(JOPNT,2).OR.YCROS.LT.TRCPT(JOPNT,4) )
     *            GO TO 110
        ENDIF
C
        TRITH = SQRT( (XMEAN-XCROS)**2 + (YMEAN-YCROS)**2 )
        IF ( TRITH.GT.TDSTH ) GO TO 110
        XRESI = XCROS - XMEAN
        IF ( XRESI.GT.0..AND.XRESI.GT.XMAXL ) XMAXL = XRESI
        IF ( XRESI.LT.0..AND.XRESI.LT.XMINL ) XMINL = XRESI
  110   CONTINUE
      ENDIF
C
      CALL GGUBS( DSEED, 1, RSCLE )
      IF ( RSCLE(1).LE.0.232 ) THEN
         TRCPT(JOPNT,1) = XMEAN + XMINO
         TRCPT(JOPNT,3) = XMEAN + XMAXO
      ELSE
         TRCPT(JOPNT,1) = XMEAN + XMINL
         TRCPT(JOPNT,3) = XMEAN + XMAXL
      ENDIF
      TRCPT(JOPNT,2) = TSEND(JOPNT) * ( TRCPT(JOPNT,1) - XMEAN )
     *               + YMEAN
      TRCPT(JOPNT,4) = TSEND(JOPNT) * ( TRCPT(JOPNT,3) - XMEAN )
     *               + YMEAN
      TDIST(JOPNT) = SQRT( (TRCPT(JOPNT,3)-TRCPT(JOPNT,1))**2 +
     *                     (TRCPT(JOPNT,4)-TRCPT(JOPNT,2))**2 )
      IF ( TDIST(JOPNT).LT.TRUNL ) THEN
           JOPNT = JOPNT - 1
           GO TO 30
      ENDIF
C
      IF ( JOPNT.LT.NOLPT ) GO TO 30
C
      WRITE(6,900)
      DO 45 KOPNT = 1, NOFPT
      WRITE(6,910) (CORFP(KOPNT,I), I = 1, 4)
   45 CONTINUE
      WRITE(6,900)
      DO 40 KOPNT = 1, NOLPT
      WRITE(6,910) (TRCPT(KOPNT,I), I = 1, 4)
   40 CONTINUE
C
  900 FORMAT(//,5X,'SIMULATIONS OF THE TRACE LENGTHES',//)
  910 FORMAT(   5X,2(2(F15.7,5X),/,5X) )
```

```
      STOP
      END
```

Appendix D
USER MANUAL AND LIST FOR "TRACESIM"

D.1 Introduction

Program TRACESIM is used to analyze a blocky rock mass behavior. For this, the fracture pattern is first generated in a rock slope, followed by a topological analysis of fully persistent rock blocks. Finally, the stability analysis for a given rock block is performed using Coulomb's failure criterion. Output data can be stored on a tape so that plotting of the results is possible.

Program TRACESIM can be run both on VMS and on Unix operating systems. In the case of VMS system, since one can attach a random number generator such as IMSL mathematical package, minor changes in the program TRACESIM are needed (in subroutines HPOIS, NHPOI and PATEN).

D.2 Input Manual for Program TRACESIM

Free format is used for all input data except for the title. According to the input option (MOPTN), maximum three plotting data files are generated to plot the fracture pattern, effective fractures and kinetically unstable rock blocks.

1. TITLE (20A4)
2. Analysis option data : NPOIN, NOSET, NOSIM, MOPTN, NSTAT, NCASE, NSYST

    - NPOIN : No. of fractures generated in a slope
    - NOSET : No. of fracture sets in a slope
    - NOSIM : No. of simulations for a given data
    - MOPTN : Printout option

a. MOPTN = 1 : Standard output of the result

b. MOPTN = 2 : Store simulated fracture pattern in file 7

c. MOPTN = 3 : Store effective fractures and intersection points in file 8

d. MOPTN = 4 : Store cohesional and frictional forces of rock blocks in file 10

e. MOPTN = 5 : Store simulated fracture pattern, effective fractures and kinetically unstable rock blocks in files 7, 8 and 9, respectively.

- NSTAT : Option for statistical analysis

a. NSTAT = 0 : Do not consider

b. NSTAT = 1 : Store the number of kinematically admissible rock blocks according to volume.

c. NSTAT = 2 : Store the number of kinematically admissible rock blocks according to dip angle.

- NCASE : No. of analysis cases

- NSYST : Operating system

a. NSYST = 1 : VMS system

b. NSYST = 2 : Unix system

3. Read statistical analysis data (skip if NSTAT = 0) : VARMN, DSTEP, NSTEP, NREG1, NREG2

- VARMN : Minimum volume or dip angle according to NSTAT

- DSTEP : Size of increment

- NSTEP : Total no. of increments

- NREG1 : Maximum number of simulations used to store the data in the plotting file (NREG1 ≤ NOSIM)

- NREG2 : Maximum number of data case used to store the data in the plotting file (NREG2 ≤ NCASE)

4. Data for slope geometry : ANGLE, HEIGHT, RANGE, WEIGT, SFMAX (use any consistent units)

- ANGLE : Slope angle in degrees

- HEIGHT : Slope height

- RANGE : Distance between slope toe and maximum slope face (see Fig. 6-2 in the text)

- WEIGT : Unit weight of rock material

- SFMAX : Safety factor used in the kinetic analysis

5. Read geometric option and mechanical properties of rock material (Total no. of data sets = NOSET) : IOSET, NPROP(IOSET,4), PROPT(IOSET,7), DSEED, STAND (use any consistent units)

   - IOSET : No. of fracture sets
   - NPROP(IOSET,1) : No. of fractures generated
   - NPROP(IOSET,2) : Options for midpoint models of fractures
     a. NPROP(IOSET,2) = 1 : Homogeneous Poisson point process model

     b. NPROP(IOSET,2) = 2 : Non-homogeneous Poisson point process model
   - NPROP(IOSET,3) : Options for trace length distribution models
     a. NPROP(IOSET,3) = 1 : Exponential distributrion model

     b. NPROP(IOSET,3) = 2 : Lognormal distribution model
   - NPROP(IOSET,4) : Options for fracture orientation models
     a. NPROP(IOSET,4) = 1 : Von Mises distribution model

     b. NPROP(IOSET,4) = 2 : Wrapped normal distribution model (currently ignored)

     c. NPROP(IOSET,4) = 3 : Uniform distribution model

     d. NPROP(IOSET,4) = 4 : Fixed (i.e., parallel) orientation model
   - PROPT(IOSET,1) : Mean trace length
   - PROPT(IOSET,2) : Standard deviation of trace length (input 0 when NPROP(IOSET,3) = 1)
   - PROPT(IOSET,3) : Mean fracture orientation in degrees
   - PROPT(IOSET,4) : Concentration factor of the Von Mises orientation distribution. For other distribution, i.e., NPROP(IOSET,4) = 2, 3 or 4, set PROPT(IOSET,4) = 0.
   - PROPT(IOSET,5) : Cohesion of fracture
   - PROPT(IOSET,6) : Friction angle of fracture
   - PROPT(IOSET,7) : Tensile cut-off stress of fracture
   - Read random number seed : DSEED or ISEED (input integer value when NSYST = 2)
   - Input max. radius of influence when NPROP(IOSET,2) = 2 : STAND

D.3 Program Listing : TRACESIM

```
      PROGRAM TRACESIM
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               C
C     PROGRAM TRACESIM IS INITIALLY INTENDED TO SIMULATE THE    C
C     MECHANICAL FRACTURING BEHAVIOR OF JOINTS ON A SLOPE       C
C     WHICH INCLUDES WING CRACK, COALESCENT CRACK AND SECOND    C
C     CRACK USING STOCHASTIC JOINT GEOMETRY MODEL.              C
C                                                               C
C     VERSION 1 IS DEVELOPED  IN THE ROCK MECHANICS GROUP AT    C
C     MIT, FEB. 1990, USING JVNC SUPER COMPUTER.                C
C     THIS VERSION IS USED FOR THE SLOPE STABILITY ANALYSIS     C
C     CONSIDERING FULLY PERSISTENT ROCK BLOCK MODEL             C
C                                                               C
C     VERSION 1.1 IS  UPDATED AND ADAPTED TO UNIX SYSTEM        C
C                                                               C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *              NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      COMMON/CONTOL/NMODE, SFMIN, MOSIM
      COMMON/STATIC/NSTAT, VARMN, DSTEP, NSTEP, ICASE, NREG1, NREG2
      DIMENSION KK(10000), AA(40000), TITLE(20)
C
      NRVAR = 40000
      NIVAR = 10000
      NDOFN = 2
C
C*** OPEN STATEMENT
C
      OPEN (5, FILE='tracesim.dat5', STATUS='UNKNOWN' )
      OPEN (6, FILE='slope.out6', STATUS='UNKNOWN' )
C
      READ (5,900) TITLE
      WRITE(6,900) TITLE
C
C     1.  NO. OF SIMULATED JOINTS IN A SLOPE : NPOIN
C     2.  NO. OF JOINT SETS  : NOSET
C     3.  NO. OF SIMULATIONS : NOSIM
C
C     4.  PRINTOUT OPTION    : MOPTN
C         1 = PRINT A SIMULATED TRACE PATTERN IN STANDARD OUTPUT FILE 6
C         2 = PRINT A SIMULATED TRACE PATTERN BOTH
C             IN STANDARD OUTPUT FILE 6 AND IN PLOTTING FILE 7
C         3 = PRINT A DETECTED TRACE PATH IN PLOTTING FILE 8
C         4 = STORE SAFETY FACTOR IN FILE 10
C         5 = STORE ALL DATA IN PLOTTING FILES 7, 8 AND 9
C
C     5.  OPTION FOR STATISTICAL ANALYSIS : NSTAT
C         0 = DO NOT CONSIDER
C         1 = STATISTICAL ANALYSIS OF S.F. ACCORDING TO VOLUME
```

```
C         2 = STATISTICAL ANALYSIS OF S.F. ACCORDING TO DIP
C
C     6.  OPTION FOR NO. OF CASES : NCASE
C     7.  OPTION FOR OPERATING SYSTEM : NSYST
C         1 = RUNNING ON VMS SYSTEM
C         2 = RUNNING ON Unix SYSTEM
C
C     8.  ANALYSIS OPTION : NOPTN
C
C         1 = HOMOGENEOUS POISSON MID-POINT PATTERN
C         2 = NON-HOMOGENEOUS POISSON MID-POINT PATTERN
C
C     9.  TRACE LENGTH DISTRIBUTION OPTION : NOPLT
C
C         1 = EXPONENTIAL DISTRIBUTION WITH MEAN TRACE LENGTH
C         2 = LOGNORMAL DISTRIBUTION WITH MEAN AND STD. DEVIATION
C
C    10.  ORIENTATION  DISTRIBUTION OPTION : NOPDP
C
C         1 = Von Mises DISTRIBUTION FUNCTION WITH MEAN ORIENTATION
C             AND CONCENTRATION FACTOR
C         2 = WRAPPED NORMAL DISTRIBUTION FUNCTION WITH MEAN
C             ORIENTATION AND CONCENTRATION FACTOR
C         3 = UNIFORM DISTRIBUTION ON [0, 180 deg]
C         4 = FIXED ORIENTATION ON THETA
C
C    11.  NO. OF EFFECTIVE JOINTS IN A SLOPE : NTRCE
C
C    12.  PRESCRIBED SAFETY FACTOR : SFMAX
C
      READ (5,*) NPOIN, NOSET, NOSIM, MOPTN, NSTAT, NCASE, NSYST
      WRITE(6,910) NPOIN, NOSET, NOSIM, MOPTN, NSTAT, NCASE, NSYST
      MOSIM = NOSIM + 1
C
C ** OPEN STATEMENTS
C
      IF ( MOPTN.GE.2 ) OPEN (7, FILE='slope.out7', STATUS='UNKNOWN' )
      IF ( MOPTN.GE.3 ) OPEN (8, FILE='slope.out8', STATUS='UNKNOWN' )
      IF ( NSTAT.GT.0 ) OPEN (9, FILE='slope.out9', STATUS='UNKNOWN' )
      IF ( NSTAT.GT.0 ) OPEN (10,FILE='slope.out0', STATUS='UNKNOWN' )
C
C     READ PARAMETERS FOR STATISTICAL ANALYSIS
C        VARMN : MINIMUM VALUE OF VOLUME OR DIP ANGLE
C        DSTEP : SIZE OF INCREMENT
C        NSTEP : NO. OF INCREMENTS
C
      IF ( NSTAT.NE.0 ) THEN
           READ (5,*) VARMN, DSTEP, NSTEP, NREG1, NREG2
           WRITE(6,920) VARMN, DSTEP, NSTEP
      ENDIF
C
      READ (5,*) ANGLE, HEIGT, RANGE, WEIGT, SFMAX
      WRITE(6,930) ANGLE, HEIGT, RANGE, WEIGT, SFMAX
C
      MOPNR = 4
      MPARA = 7
      NOPNR = NOSET * MOPNR
      NTOTL = NDOFN * NPOIN
      NTOVL = NTOTL * 2
      NPARA = NOSET * MPARA
      NTLST = MOSIM * NSTEP
      MSTOR = NCASE * NSTEP
      IF ( NSTAT.EQ.0 ) THEN
         NTLST = 1
         MSTOR = 1
      ENDIF
C
C     DYNAMIC DIMENSIONING
C
C      NR1  : OUTPUT OF STABILITY PROBABILITY 1 = CASEO(NSTEP,NCASE)
C      NR2  : OUTPUT OF STABILITY PROBABILITY 2 = CASET(NSTEP,NCASE)
C
      NR1  = 1
      NR2  = NR1  + MSTOR
      NR3  = NR2  + MSTOR
      JVARI = NR3 - 1
      DO 1 IVARI = 1, JVARI
      AA(IVARI) = 0.
  1   CONTINUE
C
C     DO LOOP FOR EACH PARAMETER CASE
C
      DO 500 ICASE = 1, NCASE
C
C     DYNAMIC DIMENSIONING
C
C      NR3  : MATERIAL PARAMETERS = PROPT(NOSET,MPARA)
C      NR4  : KINEMATIC UNSTABLE JOINT PATHS = SVALU(MOSIM,NSTEP)
C              ( OPTIONAL )
C      NR5  : MECHANICAL UNSTABLE JOINT PATHS = TVLAU(MOSIM,NSTEP)
C              ( OPTIONAL )
C      NR6  : MID POINTS DIMENSION = COORM(NPOIN,2)
C      NR7  : END POINTS DIMENSION = COORE(NTOTL,2)
C
      NR4    = NR3 + NPARA
      NR5    = NR4 + NTLST
      NR6    = NR5 + NTLST
      NR7    = NR6 + NTOTL
      NR8    = NR7 + NTOVL
C
C      NI1  : NO. OPTION PARAMETERS IN TOTAL JOINT SETS = NPROP(NOSET,4)
C      NI2  : NO. INTERSECTION POINT AT EACH TRACE = NINPT(NPOIN)
C
      NI1    = 1
      NI2    = NI1 + NOPNR
      NI3    = NI2 + NPOIN
C
      LVAR1 = NR6 - 1
      LVAR2 = NI2 - 1
      DO 5 IVARI = NR3, LVAR1
```

```
 5    AA(IVARI) = 0.
      DO 6 IVARI = 1, LVAR2
 6    KK(IVARI) = 0

C
C     MAIN LOOP
C
      DO 100 IOSIM = 1, NOSIM
C
C     INITIALZE ARRAYS
C
      DO 10 IVARI = NR6, NRVAR
 10   AA(IVARI) = 0.
      DO 20 IVARI = NI2, NIVAR
 20   KK(IVARI) = 0
C
C     CALL MAIN OPTION
C
      CALL MAINS ( AA(NR3 ), AA(NR6 ), AA(NR7 ), KK(NI1 ), KK(NI2 ),
     *             MAXPT, NOEFJ, MAXBT )
C
      IF ( NOEFJ.EQ.0.OR.MAXBT.LT.2 ) THEN
         WRITE(6,980) NOEFJ, MAXBT
         GO TO 100
      ENDIF
C
C     CALL GEOMETRIC SCREENING OPTION
C
      NTRCE = NOEFJ + 2
      NTOCY = NTRCE * NTRCE
      NTOCO = NOEFJ * 4
C
C      NR8    : TRACE CONNECTIVITY MATRIX = XCONN(NTRCE,NTRCE)
C      NR9    : TRACE CONNECTIVITY MATRIX = YCONN(NTRCE,NTRCE)
C      NR10   : COORDINATE MATRIX OF EFFECTIVE TRACES = COORF(NOEFJ,4)
C
C      NI3    : NO. INTERSECTION POINTS AT EFECTIVE JOINT = NINEP(NOEFJ)
C      NI4    : MAERIAL PROPERTY AT EACH EFFECTIVE JOINT  = NMTRL(NOEFJ)
C
      NR9   = NR8  + NTOCY
      NR10  = NR9  + NTOCY
      NR11  = NR10 + NTOCO
C
      NI4   = NI3 + NOEFJ
      NI5   = NI4 + NOEFJ
C
      NRTOT = NR11 - 1
      NITOT = NI5  - 1
      IERRO = 0
C
      IF ( NRVAR.GT.NRTOT ) GO TO 30
      WRITE(6,950)
      IERRO = IERRO + 1
 30   CONTINUE
      IF ( NIVAR.GT.NITOT ) GO TO 40
```

```
        WRITE(6,970)
        IERRO = IERRO + 1
   40   CONTINUE
        IF ( IERRO.GT.0 ) STOP
C
        CALL KINEM ( AA(NR1 ), AA(NR2 ), AA(NR3 ), AA(NR4 ), AA(NR5 ),
     *               AA(NR6 ), AA(NR7 ), AA(NR8 ), AA(NR9 ), AA(NR10),
     *               KK(NI1 ), KK(NI2 ), KK(NI3 ), KK(NI4 ),
     *               MAXPT, NOEFJ )
C
  100   CONTINUE
  500   CONTINUE
C
  900   FORMAT(20A4)
  910   FORMAT(//, 5X, 'NO. JOINTS      IN A SLOPE   = ', 3X,      I5,
     *          /, 5X, 'NO. JOINT SETS IN A SLOPE   = ', 3X,      I5,
     *          /, 5X, 'NO. SIMULATIONS             = ', 3X,      I5,
     *          /, 5X, 'PRINTOUT OPTION             = ', 3X,      I5,
     *          /, 7X, '( 0 : NO PRINTOUT             )',
     *          /, 7X, '( 1 : STANDARD PRINTOUT       )',
     *          /, 7X, '( 2 : CREATE PLOTTING FILE 7  )',
     *          /, 7X, '( 3 : CREATE FILES 7 AND  8   )',
     *          /, 7X, '( 4 : INCLUDE PERTURBED DATA  )',
     *          /, 5X, 'STATISTICAL ANALYSIS OPTION = ', 3X,      I5,
     *          /, 7X, '( 0 : DO NOT CONSIDER         )',
     *          /, 7X, '( 1 : VOLUME CRITERION        )',
     *          /, 7X, '( 2 : MIN. DIP CRITERION      )',
     *          /, 5X, 'NO. OF SIMULATION CASES     = ', 3X,      I5,
     *          /, 5X, 'OPERATING SYSTEM            = ', 3X,      I5,
     *          /, 7X, '( 1 : RUNNING ON VMS  SYSTEM  )',
     *          /, 7X, '( 2 : RUNNING ON Unix SYSTEM  )'            )
  920   FORMAT(//, 5X, 'STATISTICAL ANALYSIS',
     *          /, 7X, 'STARTING VOLUME OR DIP    = ', F10.3,
     *          /, 7X, 'SIZE OF INCREMENT         = ', F10.3,
     *          /, 7X, 'NO. OF INCREMENTS         = ', I10  )
  930   FORMAT(//, 5X, 'SLOPE SHAPE CONFIGURATION',
     *          /, 7X, 'SLOPE ANGLE ( Degrees )   = ', F10.3,
     *          /, 7X, 'SLOPE HEIGHTS             = ', F10.3,
     *          /, 7X, 'FREE SURFACE RANGES       = ', F10.3,
     *         //, 5X, 'UNIT WEIGHT OF ROCK MASS  = ', F10.3,
     *          /, 5X, 'PRESCRIBED SAFETY FACTOR  = ', F10.3    )
  950   FORMAT(//, 5X, '*** INCREASE STORAGE FOR REAL ARRAY'     )
  970   FORMAT(//, 5X, '*** INCREASE STORAGE FOR INTEGER ARRAY'  )
  980   FORMAT(//, 5X, '*** NOT APPROPRIATE JOINT PATTERN ***',
     *          /, 7X, 'TOTAL INTERSECTION POINTS   = ', I5,
     *          /, 7X, 'TOTAL INTERSECTING BOUNDARY = ', I5)
C
        STOP
        END
CCC
C
        SUBROUTINE MAINS ( PROPT, COORM, COORE, NPROP, NINPT,
     *                     MAXPT, NOEFJ, MAXBT )
C
C       SUBROUTINE MAINS CONTROLS THE OPTIONS
```

```
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *              NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      COMMON/CONTOL/NMODE, SFMIN, MOSIM
      COMMON/STATIC/NSTAT, VARMN, DSTEP, NSTEP, ICASE, NREG1, NREG2
      DIMENSION COORM(NPOIN,2), COORE(NPOIN,4), PROPT(NOSET,7)
      DIMENSION NPROP(NOSET,4), NINPT(NPOIN)
C
C     FIRST, PROGRAM TRACESIM SIMULATES THE APPROPRIATE JOINT
C     PATTERN IN A SLOPE ACCORDING TO THE OPTIONS OF
C     NOPTN, NOPLT AND NOPDP.
C
      CALL GENES ( COORM, COORE, PROPT, NPROP )
C
C     SECOND, CONSIDER THE GEOMETRICALLY ADMISSIBLE JOINT PATHS
C
      CALL GEOTY ( COORM, COORE, NINPT, MAXPT, NOEFJ, MAXBT )
C
      RETURN
      END
CCC
C
      SUBROUTINE GENES ( COORM, COORE, PROPT, NPROP )
C
C     SUBROUTINE GENES GENERATES THE JOINT PATTERN IN A SLOPE
C     ACCORDING TO THE OPTIONS OF NOPTN, NOPLT AND NOPDP.
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *              NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      DIMENSION COORM(NPOIN,2), COORE(NPOIN,4)
      DIMENSION NPROP(NOSET,4), PROPT(NOSET,7)
C
      PIRAD = 3.1415926 / 180.
C
C     READ GEOMETRY OPTIONS
C
      IF ( IOSIM.EQ.1 ) THEN
         NTRPT = 0
         DO 10 KOSET = 1, NOSET
            READ (5,*) IOSET
            READ (5,*) ( NPROP(IOSET,JOSET),JOSET= 1, 4 )
            READ (5,*) ( PROPT(IOSET,JOSET),JOSET= 1, 7 )
            WRITE(6,910) IOSET, ( NPROP(IOSET,JOSET),JOSET= 1, 4 )
            WRITE(6,915) ( PROPT(IOSET,JOSET),JOSET= 1, 7 )
            PROPT(IOSET,3) = PROPT(IOSET,3) * PIRAD
            PROPT(IOSET,6) = PROPT(IOSET,6) * PIRAD
            NTRPT = NTRPT + NPROP(IOSET,1)
 10      CONTINUE
         IF ( NTRPT.NE.NPOIN ) THEN
            WRITE(6,920)
            RETURN
         ENDIF
      ENDIF
```

```
C
C     GENERATE THE MID-POINT PATTERN OF JOINTS IN A SLOPE : NOPTN
C
      IPOIN = 0
      DO 400 IOSET = 1, NOSET
      NOPTN = NPROP(IOSET,2)
      GO TO (100, 200), NOPTN
C
C     A HOMOGENEOUS POISSON POINT PATTERN
C
 100  CONTINUE
      CALL HPOIS ( COORM, ISEED, DSEED, NPROP, IOSET, IPOIN )
      GO TO 300
C
C     A NON-HOMOGENEOUS POISSON POINT PATTERN
C
 200  CONTINUE
      CALL NHPOI ( COORM, ISEED, DSEED, NPROP, IOSET, IPOIN )
 300  CONTINUE

C
C     GENERATE A TRACE LENGTH AND AN ORIENTATION OF A JOINT
C     AT EACH MID-POINT ACCORDING TO NOPLT AND NOPDP
C
      CALL PATEN ( COORM, COORE, ISEED, DSEED, NPROP, IOSET,
     *             PROPT, IPOIN )
 400  CONTINUE
C
 910  FORMAT(//, 5X, '*** JOINT SET NO.                       = ', I5,
     *        /, 7X, 'NO. OF JOINTS                         = ', I5,
     *        /, 7X, 'OPTION FOR JOINT PATTERN DISTRIBUTION = ', I5,
     *        /, 9X, '( 1 : HOMO. POISSON POINT PATTERN        )'  ,
     *        /, 9X, '( 2 : NON-HOMO. POISSON POINT PATTERN    )'  ,
     *        /, 7X, 'OPTION FOR JOINT LENGTH DISTRIBUTION  = ', I5,
     *        /, 9X, '( 1 : EXPONENTIAL LENGTH DISTRIBUTION    )'  ,
     *        /, 9X, '( 2 : LOGNORMAL   LENGTH DISTRIBUTION    )'  ,
     *        /, 7X, 'OPTION FOR ORIENTATION DISTRIBUTION   = ', I5,
     *        /, 9X, '( 1 : VON MISES ORIENTATION DISTRIBUTION )'  ,
     *        /, 9X, '( 2 : WRAPPED   NORMAL      DISTRIBUTION )'  ,
     *        /, 9X, '( 3 : UNIFORM   ORIENTATION DISTRIBUTION )'  ,
     *        /, 9X, '( 4 : FIXED     ORIENTATION              )'  )
 915  FORMAT(//, 7X, 'GEOMETRIC PARAMETERS OF JOINTS',
     *        /, 9X, 'MEAN JOINT LENGTH                   = ', F10.3,
     *        /, 9X, 'STD. DEVIATION (LOGNORMAL)          = ', F10.3,
     *        /, 9X, 'MEAN ORIENTATION ( Degree )         = ', F10.3,
     *        /, 9X, 'CONCENTRATION FACTOR                = ', F10.3,
     *        /, 7X, 'MECHANICAL PROPERTIES OF JOINTS',
     *        /, 9X, 'COHESION OF JOINT FACE              = ', F10.3,
     *        /, 9X, 'FRICTION ANGLE OF JOINT FACE (Deg.) = ', F10.3,
     *        /, 9X, 'TENSILE CUT-OFF STRESS OF JOINT     = ', F10.3 )
 920  FORMAT(//, 5X, '*** WARNING *** : NO. POINT NOT MATCHED' )
C
      RETURN
      END
CCC
```

```
C
      SUBROUTINE HPOIS ( COORM, ISEED, DSEED, NPROP, IOSET, IPOIN )
C
C     SUBROUTINE HPOIS GENERATES THE HOMOGENEOUS POISSON POINT
C     PATTERN IN A SLOPE
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *              NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      DIMENSION COORM(NPOIN,2), TMPRY(2)
      DIMENSION NPROP(NOSET,4)
C
C     CALL RANDOM NUMBER GENERATOR
C     ACCORDING TO UNIX OR VMS MODE
C
      IF ( IOSIM.EQ.1.AND.IOSET.EQ.1 ) THEN
         IF ( NSYST.EQ.1 ) THEN
C
C *** REAL SEED FOR VMS SYSTEM ( USE IMSL ROUTINE )
C
            READ (5,*) DSEED
         ELSE
C
C *** INTEGER SEED FOR UNIX SYSTEM FUNCTION RANDOM
C
            READ (5,* ) ISEED
         ENDIF
      ENDIF
C
      NOPNT = NPROP(IOSET,1)
      IOPNT = 0
      PIVAL = 3.1415926
      ANRAD = ANGLE * PIVAL / 180.
C
 10   CONTINUE
      IF ( NSYST.EQ.2 ) THEN
C
C *** CALL INTRINSIC FUNCTION IN UNIX SYSTEM
C
         TMPRY(1) = RANDOM (ISEED)
         ISEED = IRANDM (ISEED)
         TMPRY(2) = RANDOM (ISEED)
         ISEED = IRANDM (ISEED)
      ELSE
C
C *** CALL IMSL IN VMS SYSTEM
C
C         CALL GGUBS (DSEED, 2, TMPRY)
      ENDIF
C
      TMPCX = TMPRY(1) * RANGE
      TMPSP = TMPCX * TAN(ANRAD)
      TMPCY = TMPRY(2) * HEIGT
C
C     CHECK THE SLOPE FACE BOUNDARY
```

```
C
      IF ( TMPCY.GT.TMPSP ) GO TO 10
C
      IPOIN = IPOIN + 1
      IOPNT = IOPNT + 1
      COORM(IPOIN,1) = TMPCX
      COORM(IPOIN,2) = TMPCY
      IF ( IOPNT.LT.NOPNT ) GO TO 10
C
      RETURN
      END
CCC
C
      SUBROUTINE NHPOI ( COORM, ISEED, DSEED,
     *                   NPROP, IOSET, IPOIN, ICASE )
C
C     SUBROUTINE NHPOI GENERATES THE NON HOMOGENEOUS
C     HIERARCHICAL POINT PATTERN IN A SLOPE.
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *              NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      DIMENSION COORM(NPOIN,2), TMPRY(2)
      DIMENSION NPROP(NOSET,4)
C
C     CURRENTLY, CLUSTERING PATTERN OF THE SECOND SET AROUND
C     THE FIRST SET CAN BE REALIZED.
C
      NOPNT = NPROP(IOSET,1)
      IOPNT = 0
      PIVAL = 3.1415926
      ANRAD = ANGLE * PIVAL / 180.
      NOPRS = NPOIN - NOPNT
C
      IF ( IOSIM.EQ.1.AND.IOSET.NE.1 ) THEN
         READ (5,*) STAND
         WRITE(6,100) STAND
      ENDIF
C
C      DO 50 IOPNT = 1, NOPNT
 10   CONTINUE
C
      IF ( NSYST.EQ.2 ) THEN
C
C *** CALL RANDOM IN UNIX SYSTEM
C
         TMPRY(1) = RANDOM (ISEED)
         ISEED = IRANDM (ISEED)
         TMPRY(2) = RANDOM (ISEED)
         ISEED = IRANDM (ISEED)
      ELSE
C
C *** VMS SYSTEM
C
C          CALL GGUBS (DSEED, 2, TMPRY)
```

```
      ENDIF
C
      KRNDM = TMPRY(1) * NOPRS + 1
      COORX = COORM(KRNDM,1)
      COORY = COORM(KRNDM,2)
      TMPCX = COORX +  STAND * TMPRY(1)   - STAND / 2.
      TMPCY = COORY +  STAND * TMPRY(2)   - STAND / 2.
      TMPSP = TMPCX * TAN(ANRAD)
C
      IF ( TMPCX.LE.0..OR.TMPCX.GE.RANGE ) GO TO 10
      IF ( TMPCY.LE.0..OR.TMPCY.GE.HEIGT ) GO TO 10
      IF ( TMPCY.GT.TMPSP ) GO TO 10
C
      IPOIN = IPOIN + 1
      IOPNT = IOPNT + 1
      COORM(IPOIN,1) = TMPCX
      COORM(IPOIN,2) = TMPCY
      IF ( IOPNT.LT.NOPNT ) GO TO 10
C
 100  FORMAT(5X,'INFLUENCE ZONE OF CLUSTERING PATTERN = ',3X,F12.2)
C
      RETURN
      END
CCC
C
      SUBROUTINE PATEN ( COORM, COORE, ISEED, DSEED, NPROP, IOSET,
     *                   PROPT, IPOIN )
C
C     SUBROUTINE PATEN EVALUATES THE TRACE LENGTH AND ORIENTATION
C     OF A TRACE ACCORDING TO NOPLT AND NOPDP
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *              NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      COMMON/CONTOL/NMODE, SFMIN, MOSIM
      DIMENSION COORM(NPOIN,2), COORE(NPOIN,4), TMPA1(1), TMPA2(1)
      DIMENSION NPROP(NOSET,4), PROPT(NOSET,7)
C
      PIRAD = 3.1415926
      RADIN = PIRAD / 180.
      ANRAD = ANGLE * RADIN
C
      NOPNT = NPROP(IOSET,1)
      NOPLT = NPROP(IOSET,3)
      NOPDP = NPROP(IOSET,4)
C
      TRCEM = PROPT(IOSET,1)
      STDTN = PROPT(IOSET,2)
      THETA = PROPT(IOSET,3)
      CONFT = PROPT(IOSET,4)
C
C     USE MID-POINT COORDINATES EITHER FROM SUBROUTINE HPOIS OR
C     FROM NHPOI.
C     FROM IMSL, CHOOSE SUBROUTINES GGEXN OR GGNLG FOR EXPONENTIAL
C     AND LOGNORMAL TRACE LENGTH DISTRIBUTION, RESPECTIVELY.
```

```
C
      KPOIN = IPOIN - NOPNT
      JPOIN = KPOIN + 1
 10   CONTINUE
      KPOIN = KPOIN + 1
C
      IF ( NSYST.EQ.1 ) THEN
C        IF ( NOPLT.EQ.1 ) THEN
C           CALL GGEXN ( DSEED, TRCEM, 1, TMPA1 )
C        ELSE
C           CALL GGNLG ( DSEED, 1, TRCEM, STDTN, TMPA1 )
C        ENDIF
      ELSE
C
C *** EXPONENTIAL TRACE LENGTH GENERATOR WITH UNIX SYSTEM
C
         TMPA1(1) = RANDOM (ISEED)
         ISEED = IRANDM (ISEED)
         TMPA1(1) = -ALOG ( 1. - TMPA1(1) ) * TRCEM
      ENDIF
C
      TCLEN = TMPA1(1) / 2.
C
C     VON MISES ORIENTATION DISTRIBUTION
C
      IF ( NOPDP.EQ.1 ) THEN
C        xindx = 1. + ( 0.5*conft )**2 + ( 0.5*conft )**4 / 4.
C     *        + ( 0.5*conft )**6 / 36.
C        xindx = 1. / ( xindx * 2. * pirad )
C
C *** UNIX SYSTEM FUNCTION
C
         IF ( NSYST.EQ.2 ) THEN
            FREQX = EXP ( CONFT )
 99         CONTINUE
            TMPA2(1) = RANDOM (ISEED)
            ISEED = IRANDM (ISEED)
            YINDX = TMPA2(1) * PIRAD * 2.
            FREQY = EXP ( CONFT * COS( YINDX - THETA ) )
            FREQZ = FREQY / FREQX
            TMPA2(1) = RANDOM (ISEED)
            ISEED = IRANDM (ISEED)
            IF ( TMPA2(1).GT.FREQZ ) GO TO 99
            ORINT = YINDX
         ELSE
C
C ***  VMS MODE
C
C            CALL GGVMS ( DSEED, CONFT, 1, TMPA2 )
            ORINT = TMPA2(1) + THETA
         ENDIF
      ENDIF
C
C     CURRENTLY (NOPDP.EQ.2) OPTION (WRAPPED NORMAL DISTRIBUTION)
C     IS IGNORED
```

```
C       IF ( NOPDP.EQ.2 ) THEN
C
C     UNIFORM ORIENTATION DISTRIBUTION
C
      IF ( NOPDP.EQ.3 ) THEN
         IF ( NSYST.EQ.1 ) THEN
C
C *** VMS
C
C             CALL GGUBS ( DSEED, 1, TMPA2 )
         ELSE
C
C *** UNIX SYSTEM FUNCTION
C
            TMPA2(1) = RANDOM (ISEED)
            ISEED = IRANDM (ISEED)
         ENDIF
         ORINT = TMPA2(1) * PIRAD
      ENDIF
C
C     FIXED ORIENTATION
C
      IF ( NOPDP.EQ.4 ) ORINT = THETA
C
C     CALCULATE END POINTS OF EACH TRACE
C
      XCOOR = COORM(KPOIN,1)
      YCOOR = COORM(KPOIN,2)
      XCOR1 = XCOOR + TCLEN * COS(ORINT)
      YCOR1 = YCOOR + TCLEN * SIN(ORINT)
      XCOR2 = XCOOR - TCLEN * COS(ORINT)
      YCOR2 = YCOOR - TCLEN * SIN(ORINT)
C
C     CONSIDER THE BOUNDARY CONDITIONS
C
      IF ( YCOR1.GE.HEIGT ) THEN
         YCOR1 = HEIGT
         IF ( ORINT.EQ.PIRAD/2. ) THEN
            XCOR1 = XCOOR
         ELSE
            XCOR1 = XCOOR + ( YCOR1-YCOOR ) / TAN(ORINT)
         ENDIF
      ENDIF
      IF ( YCOR2.GE.HEIGT ) THEN
         YCOR2 = HEIGT
         IF ( ORINT.EQ.PIRAD/2. ) THEN
            XCOR2 = XCOOR
         ELSE
            XCOR2 = XCOOR + ( YCOR2-YCOOR ) / TAN(ORINT)
         ENDIF
      ENDIF
      IF ( YCOR1.LE.0. ) THEN
         YCOR1 = 0.
         IF ( ORINT.EQ.PIRAD/2. ) THEN
            XCOR1 = XCOOR
            ELSE
               XCOR1 = XCOOR + ( YCOR1-YCOOR ) / TAN(ORINT)
            ENDIF
         ENDIF
         IF ( YCOR2.LE.0 ) THEN
            YCOR2 = 0.
            IF ( ORINT.EQ.PIRAD/2. ) THEN
               XCOR2 = XCOOR
            ELSE
               XCOR2 = XCOOR + ( YCOR2-YCOOR ) / TAN(ORINT)
            ENDIF
         ENDIF
         IF ( XCOR1.GE.RANGE ) THEN
            XCOR1 = RANGE
            YCOR1 = TAN(ORINT) * ( XCOR1-XCOOR ) + YCOOR
         ENDIF
         IF ( XCOR1.LE.0. ) THEN
            XCOR1 = 0.
            YCOR1 = TAN(ORINT) * ( XCOR1-XCOOR ) + YCOOR
         ENDIF
         IF ( XCOR2.GE.RANGE ) THEN
            XCOR2 = RANGE
            YCOR2 = TAN(ORINT) * ( XCOR2-XCOOR ) + YCOOR
         ENDIF
         IF ( XCOR2.LE.0. ) THEN
            XCOR2 = 0.
            YCOR2 = TAN(ORINT) * ( XCOR2-XCOOR ) + YCOOR
         ENDIF
C
         SLOP1 = XCOR1 * TAN(ANRAD)
         SLOP2 = XCOR2 * TAN(ANRAD)
         IF ( YCOR1.GT.SLOP1 ) THEN
            YCOR3 = YCOOR - YCOR1
            XCOR3 = XCOOR - XCOR1
            IF ( ABS(XCOR3).LT.0.0001 ) THEN
               YCOR1 = TAN(ANRAD) * XCOOR
            ELSE
               TANTC = YCOR3 / XCOR3
               XCOR1 = ( TANTC * XCOR1 - YCOR1 ) /
        *          ( TANTC - TAN(ANRAD) )
               YCOR1 = TAN(ANRAD) * XCOR1
            ENDIF
         ENDIF
         IF ( YCOR2.GT.SLOP2 ) THEN
            YCOR4 = YCOOR - YCOR2
            XCOR4 = XCOOR - XCOR2
            IF ( ABS(XCOR4).LT.0.0001 ) THEN
               YCOR2 = TAN(ANRAD) * XCOOR
            ELSE
               TANTC = YCOR4 / XCOR4
               XCOR2 = ( TANTC * XCOR2 - YCOR2 ) /
        *          ( TANTC - TAN(ANRAD) )
               YCOR2 = TAN(ANRAD) * XCOR2
            ENDIF
         ENDIF
```

```
C
C     STORE THE END POINTS OF EACH TRACE INTO COORE
C
      COORE(KPOIN,1) = XCOR1
      COORE(KPOIN,2) = YCOR1
      COORE(KPOIN,3) = XCOR2
      COORE(KPOIN,4) = YCOR2
      IF ( XCOR1.GT.XCOR2 ) THEN
         COORE(KPOIN,1) = XCOR2
         COORE(KPOIN,2) = YCOR2
         COORE(KPOIN,3) = XCOR1
         COORE(KPOIN,4) = YCOR1
      ENDIF
C
      IF ( KPOIN.LT.IPOIN ) GO TO 10
C
C     IF ( MOPTN.GE.1 ), WRITE THE END POINT COORDINATES
C
      IF ( MOPTN.EQ.1 ) THEN
         WRITE(6,900) IOSET
         DO 20 KPOIN = JPOIN, NPOIN
            WRITE(6,910) ( COORE(KPOIN,JDOFN),JDOFN = 1,4 )
 20      CONTINUE
      ENDIF
      IF ( MOPTN.GE.5 ) THEN
         IF ( IOSIM.EQ.1.AND.IOSET.EQ.1 )
     *         WRITE(7,920) NPOIN, ANGLE, HEIGT, RANGE, NOSIM, MOPTN
         DO 30 KPOIN = JPOIN, IPOIN
            WRITE(7,910) ( COORE(KPOIN,JDOFN),JDOFN = 1,4 )
 30      CONTINUE
      ENDIF
C
 900  FORMAT(//,5X,'SIMULATED TRACE PATTERN OF JOINT SET = ', I5,//)
 910  FORMAT(2(5X,F10.3,5X,F10.3,/))
 920  FORMAT(I5, 3F10.3, I5)
C
      RETURN
      END
CCC
C
      SUBROUTINE GEOTY ( COORM, COORE, NINPT, MAXPT, NOEFJ, MAXBT )
C
C     SUBROUTINE GEOTY EVALUATES THE GEOMETRICALLY
C     CONTINUOUS JOINT PATHS
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *               NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      DIMENSION COORM(NPOIN,2), COORE(NPOIN,4), NINPT(NPOIN)
C
C     TO FIND THE EFFECTIVE JOINT PATHS, CALCULATE THE EFFECTIVE
C     INTERSECTION POINTS AMONG JOINTS AND SLOPE BOUNDARIES
C
      PIRAD = 3.1415926
      TANAN = TAN(ANGLE*PIRAD / 180.)
      XAPEX = HEIGT / TANAN
      MAXPP = 0
      IITER = 0
C
 5    CONTINUE
      MAXPT = 0
      MAXBT = 0
      NOEFJ = 0
      IITER = IITER + 1
C
      DO 30 IPOIN = 1, NPOIN
      IF ( IITER.GT.1.AND.NINPT(IPOIN).LT.2 ) GO TO 30
      NINPT(IPOIN) = 0
      XCOR1 = COORE(IPOIN,1)
      YCOR1 = COORE(IPOIN,2)
      XCOR2 = COORE(IPOIN,3)
      YCOR2 = COORE(IPOIN,4)
      IF ( XCOR2.EQ.XCOR1 ) THEN
         SLOP1 = 9.E+10
      ELSE
         SLOP1 = ( YCOR2-YCOR1 ) / ( XCOR2-XCOR1 )
      ENDIF
C
      DO 10 JPOIN = 1, NPOIN
      IF ( JPOIN.EQ.IPOIN ) GO TO 10
      IF ( IITER.GT.1.AND.NINPT(JPOIN).LT.2 ) GO TO 10
      XCOR3 = COORE(JPOIN,1)
      YCOR3 = COORE(JPOIN,2)
      XCOR4 = COORE(JPOIN,3)
      YCOR4 = COORE(JPOIN,4)
C
C     CALCULATE THE INTERSECTION POINTS AND CHECK THEIR ADMISSIBILITY
C
      COEF1 = XCOR4 - XCOR3
      COEF2 = YCOR4 - YCOR3
      IF ( COEF1.EQ.0. ) THEN
         SLOP2 = 9.E+10
      ELSE
         SLOP2 = COEF2 / COEF1
      ENDIF
      IF ( SLOP1.EQ.SLOP2 ) GO TO 10
      IF ( SLOP1.EQ.9.E+10 ) THEN
         XINPT = XCOR1
         IF (XINPT.GE.XCOR3.AND.XINPT.LE.XCOR4) THEN
            YINPT = SLOP2 * ( XINPT-XCOR3 ) + YCOR3
            IF ( YINPT.GE.AMIN1(YCOR1,YCOR2).AND.
     *           YINPT.LE.AMAX1(YCOR1,YCOR2) ) THEN
               NINPT(IPOIN) = NINPT (IPOIN) + 1
               MAXPT = MAXPT + 1
               GO TO 10
            ENDIF
         ENDIF
      ENDIF
      IF ( SLOP2.EQ.9.E+10 ) THEN
         XINPT = XCOR3
         IF (XINPT.GE.XCOR1.AND.XINPT.LE.XCOR2 ) THEN
            YINPT = SLOP1 * ( XINPT-XCOR1 ) + YCOR1
            IF ( YINPT.GE.AMIN1(YCOR3,YCOR4).AND.
     *           YINPT.LE.AMAX1(YCOR3,YCOR4) ) THEN
               NINPT(IPOIN) = NINPT(IPOIN) + 1
               MAXPT = MAXPT + 1
               GO TO 10
            ENDIF
         ENDIF
      ENDIF
C
      COEF3 = SLOP1 * XCOR1 - SLOP2 * XCOR3
      COEF4 = YCOR3 - YCOR1
      XINPT = ( COEF3 + COEF4 ) / ( SLOP1 - SLOP2 )
      YINPT = SLOP1 * (XINPT-XCOR1) + YCOR1
C
      IF ( XINPT.LT.XCOR1.OR.XINPT.GT.XCOR2 ) GO TO 10
      IF ( XINPT.LT.XCOR3.OR.XINPT.GT.XCOR4 ) GO TO 10
      IF ( YINPT.LT.AMIN1(YCOR1, YCOR2).OR.
     *     YINPT.GT.AMAX1(YCOR1, YCOR2) )     GO TO 10
      IF ( YINPT.LT.AMIN1(YCOR3, YCOR4).OR.
     *     YINPT.GT.AMAX1(YCOR3, YCOR4) )     GO TO 10
      IF ( YINPT.GT.HEIGT ) GO TO 10
C
      NINPT(IPOIN) = NINPT(IPOIN) + 1
      MAXPT = MAXPT + 1
 10   CONTINUE
C
C     SEARCH FOR INTERSECTION POINTS WITH SLOPE BOUNDARIES
C
      IF (  AMAX1(YCOR1,YCOR2).EQ.HEIGT ) THEN
         NINPT(IPOIN) = NINPT(IPOIN) + 1
         MAXBT = MAXBT + 1
      ENDIF
      IF ( SLOP1.EQ.TANAN ) GO TO 20
      IF ( SLOP1.EQ.9.E+10 ) THEN
         XINPT = XCOR1
         YINPT = TANAN * XINPT
      ELSE
         XINPT = ( SLOP1*XCOR1 - YCOR1 ) / ( SLOP1 - TANAN )
         YINPT = TANAN * XINPT
      ENDIF
      IF ( XINPT.EQ.0..AND.YINPT.EQ.0. ) THEN
         XINPT = 1.E-10
         YINPT = 1.E-10
      ENDIF
      IF ( ABS(XINPT-XCOR1).GT.0.01.AND.
     *     ABS(XINPT-XCOR2).GT.0.01 ) GO TO 20
      IF ( ABS(YINPT-AMIN1(YCOR1,YCOR2)).GT.0.01.AND.
     *     ABS(YINPT-AMAX1(YCOR1,YCOR2)).GT.0.01 ) GO TO 20
      IF ( YINPT.GT.HEIGT ) GO TO 20
      NINPT(IPOIN) = NINPT(IPOIN) + 1
      MAXBT = MAXBT + 1
C
C     COUNT THE EFFECTIVE TRACES
```

```
C
 20   CONTINUE
      IF ( NINPT(IPOIN).GE.2 ) NOEFJ = NOEFJ + 1
 30   CONTINUE
C
      IF ( IITER.EQ.1 ) WRITE(6,900) IOSIM
C
      MAXPT = MAXPT / 2 + MAXBT
C
C     FIND FURTHER ELIMINATIONS
C
      IF ( MAXPT.EQ.MAXPP ) GO TO 40
      MAXPP = MAXPT
      IF ( IITER.GT.10 ) THEN
         WRITE(6,930)
      ENDIF
      GO TO 5
C
 40   CONTINUE
C
C     DATA PORTHOLE
C
      IF ( MOPTN.GE.5 ) THEN
         WRITE(7,950) NOEFJ
         DO 50 IPOIN = 1, NPOIN
         IF ( NINPT(IPOIN).LT.2 ) GO TO 50
         WRITE(7,940) ( COORE(IPOIN,JDOFN),JDOFN = 1,4 )
 50      CONTINUE
      ENDIF
C
 900  FORMAT(//, 3X, ' *** SIMULATION NUMBER ***                 = ', I5)
 930  FORMAT(//, 5X, '*** TOO MANY ITERATIONS ***')
 940  FORMAT(2(5X,F10.3,5X,F10.3,/))
 950  FORMAT(I5)
C
      RETURN
      END
CCC
C
      SUBROUTINE KINEM ( CASEO, CASET, PROPT, SVALU, TVALU, COORM,
     *                   COORE, XCONN, YCONN, COORF, NPROP,
     *                   NINPT, NINEP, NMTRL, MAXPT, NOEFJ )
C
C     SUBROUTINE KINEM EVALUATES THE TOPOLOGICALLY AND KINEMATICALLY
C     ADMISSIBLE JOINT PATHS USING MATRIX CONNECTIVITY METHOD
C
C     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *              NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      COMMON/CONTOL/NMODE, SFMIN, MOSIM
      COMMON/CNTROL/CRITX(10), CRITY(10), NCRIT, YCRIT
      COMMON/CONROL/NCTMX(10,2)
      COMMON/STATIC/NSTAT, VARMN, DSTEP, NSTEP, ICASE, NREG1, NREG2
      DIMENSION COORM(NPOIN,2), COORE(NPOIN,4), PROPT(NOSET,4)
      DIMENSION XCONN(NTRCE,NTRCE), YCONN(NTRCE,NTRCE)
      DIMENSION CCTMX(10), CCTMY(10)
      DIMENSION MTMP1(10), MTMP2(10)
      DIMENSION COORF(NOEFJ,4), NMTRL(NOEFJ)
      DIMENSION NPROP(NOSET,4), NINPT(NPOIN), NINEP(NOEFJ)
      DIMENSION SVALU(MOSIM,NSTEP), TVALU(MOSIM,NSTEP)
      DIMENSION CASEO(NCASE,NSTEP), CASET(NCASE,NSTEP)
      DIMENSION PROBY(30), TPRBY(100)
C
C     FIRST, CALCULATES THE INTERSECTION POINTS AMONG JOINTS AND
C     SLOPE BOUNDARY USING EFFECTIVE TRACES

      PIRAD = 3.1415926
      TANAN = TAN(ANGLE*PIRAD/180.)
      NOMTX = 0
      KPOIN = 0
C
      DO 5 ITRCE = 1, NTRCE
      DO 5 JTRCE = 1, NTRCE
      XCONN(ITRCE,JTRCE) = 0.
      YCONN(ITRCE,JTRCE) = 0.
 5    CONTINUE
C
      JCONT = 0
      DO 6 IOSET = 1, NOSET
      NOPNT = NPROP(IOSET,1)
      JCONT = JCONT + NOPNT
      MTMP1(IOSET) = JCONT
      MTMP2(IOSET) = JCONT - NOPNT + 1
 6    CONTINUE
C
      DO 420 IPOIN = 1, NPOIN
      IF ( NINPT(IPOIN).GE.2 ) THEN
         KPOIN = KPOIN + 1
         NINEP(KPOIN) = NINPT(IPOIN)
C
         DO 400 IOSET = 1, NOSET
            IF ( IPOIN.GE.MTMP2(IOSET).AND.IPOIN.LE.MTMP1(IOSET) )
     *           NMTRL(KPOIN) = IOSET
 400     CONTINUE
C
         DO 410 IDOFN = 1, 4
            COORF(KPOIN,IDOFN) = COORE(IPOIN,IDOFN)
 410     CONTINUE
      ENDIF
 420  CONTINUE
C
      KPOIN = 1
      DO 50 IPOIN = 1, NOEFJ
      ICOUN = 0
      KPOIN = KPOIN + 1
      XCOR1 = COORF(IPOIN,1)
      YCOR1 = COORF(IPOIN,2)
      XCOR2 = COORF(IPOIN,3)
      YCOR2 = COORF(IPOIN,4)
      IF ( XCOR1.EQ.XCOR2 ) THEN
```

```
         SLOP1 = 9.E+10
      ELSE
         SLOP1 = ( YCOR2 - YCOR1 ) / ( XCOR2 - XCOR1 )
      ENDIF
C
C     FIRST, CALCULATE INTERSECTION POINT COORD. (XINPT, YINPT)
C     WITH SLOPE FACE
C
      IF ( SLOP1.NE.TANAN ) THEN
         IF ( SLOP1.EQ.9.E+10 ) THEN
            XINPT = XCOR1
            YINPT = TANAN * XINPT
         ELSE
            XINPT = ( SLOP1*XCOR1 - YCOR1 ) / ( SLOP1-TANAN )
            YINPT = TANAN * XINPT
         ENDIF
         IF ( XINPT.EQ.0..AND.YINPT.EQ.0.) THEN
            XINPT = 1.E-10
            YINPT = 1.E-10
         ENDIF
         IF ( ABS(XINPT-XCOR1).GT.0.01.AND.
     *        ABS(XINPT-XCOR2).GT.0.01 ) GO TO 10
         IF ( ABS(YINPT-AMIN1(YCOR1,YCOR2)).GT.0.01.AND.
     *        ABS(YINPT-AMAX1(YCOR1,YCOR2)).GT.0.01 ) GO TO 10
         IF ( YINPT.GT.HEIGT.OR.YINPT.LT.0. ) GO TO 10
C
         XCONN(1,KPOIN)  = XINPT
         XCONN(KPOIN,1)  = XINPT
         YCONN(1,KPOIN)  = YINPT
         YCONN(KPOIN,1)  = YINPT
         NOMTX = NOMTX + 2
         ICOUN = ICOUN + 1
      ENDIF
 10   CONTINUE
      LPOIN = 1
C
C     CALCULATE THE INTERSECTION POINTS AMONG JOINTS
C
      DO 30 JPOIN = 1, NOEFJ
      LPOIN = LPOIN + 1
      IF ( JPOIN.EQ.IPOIN ) GO TO 30
      XCOR3 = COORF(JPOIN,1)
      YCOR3 = COORF(JPOIN,2)
      XCOR4 = COORF(JPOIN,3)
      YCOR4 = COORF(JPOIN,4)
C
C     CALCULATE THE INTERSECTION COORDINATES AND CHECK THEIR
C     ADMISSIBILITY
C
      COEF1 = XCOR4 - XCOR3
      COEF2 = YCOR4 - YCOR3
      IF ( COEF1.EQ.0. ) THEN
         SLOP2 = 9.E+10
      ELSE
         SLOP2 = COEF2 / COEF1
      ENDIF
      IF ( SLOP1.EQ.SLOP2 ) GO TO 30
      IF ( SLOP1.EQ.9.E+10 ) THEN
         XINPT = XCOR1
         IF (XINPT.GE.XCOR3.AND.XINPT.LE.XCOR4 ) THEN
            YINPT = SLOP2 * ( XINPT-XCOR3 ) + YCOR3
            IF ( YINPT.GE.AMIN1(YCOR1,YCOR2).AND.
     *       YINPT.LE.AMIN1(YCOR1,YCOR2).AND.YINPT.LE.HEIGT ) GO TO 20
         ENDIF
         GO TO 30
      ENDIF
      IF ( SLOP2.EQ.9.E+10 ) THEN
         XINPT = XCOR3
         IF ( XINPT.GE.XCOR1.AND.XINPT.LE.XCOR2 ) THEN
            YINPT = SLOP1 * ( XINPT-XCOR1 ) + YCOR1
            IF ( YINPT.GE.AMIN1(YCOR3,YCOR4).AND.
     *       YINPT.LE.AMAX1(YCOR3,YCOR4).AND.YINPT.LE.HEIGT ) GO TO 20
         ENDIF
         GO TO 30
      ENDIF
C
      COEF3 = SLOP1 * XCOR1 - SLOP2 * XCOR3
      COEF4 = YCOR3 - YCOR1
      XINPT = ( COEF3 + COEF4 ) / ( SLOP1 - SLOP2 )
      YINPT = SLOP1 * ( XINPT-XCOR1 ) + YCOR1
C
      IF ( XINPT.LT.XCOR1.OR.XINPT.GT.XCOR2 ) GO TO 30
      IF ( XINPT.LT.XCOR3.OR.XINPT.GT.XCOR4 ) GO TO 30
      IF ( YINPT.LT.AMIN1(YCOR1,YCOR2).OR.
     *     YINPT.GT.AMAX1(YCOR1,YCOR2) ) GO TO 30
      IF ( YINPT.LT.AMIN1(YCOR3,YCOR4).OR.
     *     YINPT.GT.AMAX1(YCOR3,YCOR4) ) GO TO 30
      IF ( YINPT.GT.HEIGT ) GO TO 30
C
 20   CONTINUE
      NOMTX = NOMTX + 1
      ICOUN = ICOUN + 1
      XCONN(KPOIN,LPOIN) = XINPT
      YCONN(KPOIN,LPOIN) = YINPT
 30   CONTINUE
C
C     CALCULATE THE INTERSECTION COORDINATE WITH FREE SURFACE
C
      IF ( AMAX1(YCOR1,YCOR2).NE.HEIGT ) GO TO 40
      IF ( SLOP1.EQ.9.E+10 ) THEN
            XINPT = XCOR1
      ELSE
            XINPT = XCOR1 + ( HEIGT-YCOR1 ) / SLOP1
      ENDIF
C
      IF ( ABS(XINPT-XCOR1).GT.0.01.AND.
     *     ABS(XINPT-XCOR2).GT.0.01 ) GO TO 40
      YINPT = HEIGT
      XCONN(KPOIN,NTRCE) = XINPT
      XCONN(NTRCE,KPOIN) = XINPT
      YCONN(KPOIN,NTRCE) = YINPT
      YCONN(NTRCE,KPOIN) = YINPT
      NOMTX = NOMTX + 2
      ICOUN = ICOUN + 1
 40   CONTINUE
      IF ( ICOUN.NE.NINEP(IPOIN) ) WRITE(6,970) ICOUN, NINEP(IPOIN)
C
 50   CONTINUE
C
C     STORE THE INTERSECTION POINTS FOR PLOTTING
C
      IF ( MOPTN.GE.5 ) THEN
         MJOIN = 0
         DO 80 ITRCE = 1, NTRCE
         KTRCE = ITRCE
         DO 70 JTRCE = KTRCE, NTRCE
         IF ( XCONN(ITRCE,JTRCE).LE.0. ) GO TO 70
         WRITE(8,960) XCONN(ITRCE,JTRCE), YCONN(ITRCE,JTRCE)
         MJOIN = MJOIN + 1
 70      CONTINUE
 80      CONTINUE
C
         WRITE(7,940) MJOIN, MAXPT
      ENDIF
C
C     DO A MATRIX CONNECTIVITY SEARCHING
C     CURRENTLY, UP TO 10 (TEN) INTERSECTION POINTS CAN BE CONSIDERED
C     AS A JOINT PATH
C
      NPATR = 10
      NPATH = 0
      NMODE = 0
      LCOUN = 0
      TPATH = 0.
      SFMIN = SFMAX
C
      DO 300 ICOL1 = 1, NTRCE
C
      DO 120 IPATR = 1, NPATR
      DO 110 JDOFN = 1, NDOFN
      NCTMX(IPATR,JDOFN) = 0
 110  CONTINUE
      CCTMX(IPATR) = 0.
      CCTMY(IPATR) = 0.
 120  CONTINUE
C
C     START SEARCHING : 1ST
C
      IF ( XCONN(1,ICOL1).EQ.0. ) GO TO 300
      NCTMX(1,1) = 1
      NCTMX(1,2) = ICOL1
      CCTMX(1) = XCONN(1,ICOL1)
      CCTMY(1) = YCONN(1,ICOL1)
C
C     2ND
```

```
C
      DO 290 ICOL2 = 1, NTRCE
      IF ( XCONN(ICOL1,ICOL2).EQ.0. ) GO TO 290
      IF ( XCONN(ICOL1,ICOL2).EQ.CCTMX(1) ) GO TO 290
      IF ( ICOL2.EQ.NTRCE.AND.XCONN(ICOL1,NTRCE).NE.0. ) THEN
         NCTMX(2,1) = ICOL1
         NCTMX(2,2) = NTRCE
         NPATH = NPATH + 1
         CCTMX(2) = XCONN(ICOL1,NTRCE)
         CCTMY(2) = YCONN(ICOL1,NTRCE)
         NITER = 2
         CALL KINET ( 2, CCTMX, CCTMY, KOPTN, YPERT )
         CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ, PROPT,
     *                SVALU, TVALU )
         IF ( JFLAG.EQ.1 )
     *      CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         GO TO 290
      ENDIF
      CCTMX(2) = XCONN(ICOL1,ICOL2)
      CCTMY(2) = YCONN(ICOL1,ICOL2)
      NCTMX(2,1) = ICOL1
      NCTMX(2,2) = ICOL2
C
C *** MODIFY THE KINEMATIC JOINT PATH BY NOT ALLOWING THE DECLINING
C     2ND INTERSECTION POINT
C
      IF ( CCTMY(2).LE.CCTMY(1) ) GO TO 290
C
C     3RD
C
      DO 280 ICOL3 = 1, NTRCE
      IF ( XCONN(ICOL2,ICOL3).EQ.0.        ) GO TO 280
      IF ( XCONN(ICOL2,ICOL3).EQ.CCTMX(1) ) GO TO 280
       IF ( ICOL3.EQ.ICOL1 ) GO TO 280
      NCTMX(3,1) = ICOL2
      NCTMX(3,2) = ICOL3
      CCTMX(3) = XCONN(ICOL2,ICOL3)
      CCTMY(3) = YCONN(ICOL2,ICOL3)
      IF ( CCTMX(3).GE.CCTMX(2).AND.CCTMY(3).LE.CCTMY(2) )
     *     GO TO 280
C
      IF ( ICOL3.EQ.1.AND.CCTMX(3).NE.0. ) THEN
         IF ( XCONN(ICOL2,1).GT.CCTMX(1) ) THEN
              KOPTN = 0
              CALL KINET ( 3, CCTMX, CCTMY, KOPTN, YPERT )
              IF ( KOPTN.EQ.1 ) GO TO 280
              NPATH = NPATH + 1
              TPATH = TPATH + 1.
              NITER = -3
              CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ,
     *                     PROPT, SVALU, TVALU )
              IF ( JFLAG.EQ.1 )
     *           CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         ENDIF
         GO TO 280
      ENDIF
C
      NFLAG = 0
      IF ( CCTMY(3).LT.CCTMY(2) ) CALL KINES ( 3, CCTMX, CCTMY, NFLAG )
      IF ( NFLAG.EQ.1 ) GO TO 280
C
      IF ( ICOL3.EQ.NTRCE.AND.XCONN(ICOL2,NTRCE).NE.0. ) THEN
         KOPTN = 0
         CALL KINET ( 3, CCTMX, CCTMY, KOPTN, YPERT )
         IF ( KOPTN.EQ.1 ) GO TO 280
         NPATH = NPATH + 1
         NITER = 3
         CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ, PROPT,
     *                SVALU, TVALU )
         IF ( JFLAG.EQ.1 )
     *      CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         GO TO 280
      ENDIF
C
C     4TH
C
      DO 270 ICOL4 = 1, NTRCE
      IF ( XCONN(ICOL3,ICOL4).EQ.0.        ) GO TO 270
      IF ( XCONN(ICOL3,ICOL4).EQ.CCTMX(1) ) GO TO 270
      IF ( XCONN(ICOL3,ICOL4).EQ.CCTMX(2) ) GO TO 270
      IF ( ICOL4.EQ.ICOL1.OR.ICOL4.EQ.ICOL2 ) GO TO 270
      NCTMX(4,1) = ICOL3
      NCTMX(4,2) = ICOL4
      CCTMX(4) = XCONN(ICOL3,ICOL4)
      CCTMY(4) = YCONN(ICOL3,ICOL4)
      IF ( CCTMX(4).GE.CCTMX(3).AND.CCTMY(4).LE.CCTMY(3) )
     *     GO TO 270
C
      IF ( ICOL4.EQ.1.AND.CCTMX(4).NE.0. ) THEN
         IF ( XCONN(ICOL3,1).GT.CCTMX(1) ) THEN
            KOPTN = 0
            CALL KINET ( 4, CCTMX, CCTMY, KOPTN, YPERT )
            IF ( KOPTN.EQ.1 ) GO TO 270
            NPATH = NPATH + 1
            TPATH = TPATH + 1.
            NITER = -4
            CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ,
     *                   PROPT, SVALU, TVALU )
            IF ( JFLAG.EQ.1 )
     *         CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         ENDIF
         GO TO 270
      ENDIF
C
      NFLAG = 0
      IF ( CCTMY(4).LT.CCTMY(3) ) CALL KINES ( 4, CCTMX, CCTMY, NFLAG )
      IF ( NFLAG.EQ.1 ) GO TO 270
C
      IF ( ICOL4.EQ.NTRCE.AND.XCONN(ICOL3,NTRCE).NE.0. ) THEN
         KOPTN = 0
         CALL KINET ( 4, CCTMX, CCTMY, KOPTN, YPERT )
         IF ( KOPTN.EQ.1 ) GO TO 270
         NPATH = NPATH + 1
         NITER = 4
         CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ, PROPT,
     *                SVALU, T/ALU )
         IF ( JFLAG.EQ.1 )
     *      CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPFRT )
         GO TO 270
      ENDIF
C
C     5TH
C
      DO 260 ICOL5 = 1, NTRCE
      IF ( XCONN(ICOL4,ICOL5).EQ.0.        ) GO TO 260
      IF ( XCONN(ICOL4,ICOL5).EQ.CCTMX(1)  ) GO TO 260
      IF ( XCONN(ICOL4,ICOL5).EQ.CCTMX(2)  ) GO TO 260
      IF ( XCONN(ICOL4,ICOL5).EQ.CCTMX(3)  ) GO TO 260
      IF ( ICOL5.EQ.ICOL1.OR.
     *     ICOL5.EQ.ICOL2.OR.ICOL5.EQ.ICOL3 ) GO TO 260
      NCTMX(5,1)  = ICOL4
      NCTMX(5,2)  = ICOL5
      CCTMX(5)  = XCONN(ICOL4,ICOL5)
      CCTMY(5)  = YCONN(ICOL4,ICOL5)
      IF ( CCTMX(5).GE.CCTMX(4).AND.CCTMY(5).LE.CCTMY(4) )
     *     GO TO 260
C
      IF ( ICOL5.EQ.1.AND.CCTMX(5).NE.0. ) THEN
         IF ( XCONN(ICOL4,1).GT.CCTMX(1) ) THEN
              KOPTN = 0
              CALL KINET ( 5, CCTMX, CCTMY, KOPTN, YPERT )
              IF ( KOPTN.EQ.1 ) GO TO 260
              NPATH = NPATH + 1
              TPATH = TPATH + 1.
              NITER = -5
              CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ,
     *                     PROPT, SVALU, TVALU )
              IF ( JFLAG.EQ.1 )
     *           CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
              ENDIF
         GO TO 260
      ENDIF
C
      NFLAG = 0
      IF ( CCTMY(5).LT.CCTMY(4) ) CALL KINES ( 5, CCTMX, CCTMY, NFLAG )
      IF ( NFLAG.EQ.1 ) GO TO 260
C
      IF ( ICOL5.EQ.NTRCE.AND.XCONN(ICOL4,NTRCE).NE.0. ) THEN
         KOPTN = 0
         CALL KINET ( 5, CCTMX, CCTMY, KOPTN, YPERT )
         IF ( KOPTN.EQ.1 ) GO TO 260
         NPATH = NPATH + 1
         NITER = 5
         CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ, PROPT,
     *                SVALU, TVALU )
```

```
         IF ( JFLAG.EQ.1 )
     *      CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         GO TO 260
      ENDIF
C
C     6TH
C
      DO 250 ICOL6 = 1, NTRCE
      IF ( XCONN(ICOL5,ICOL6).EQ.0.       ) GO TO 250
      IF ( XCONN(ICOL5,ICOL6).EQ.CCTMX(1) ) GO TO 250
      IF ( XCONN(ICOL5,ICOL6).EQ.CCTMX(2) ) GO TO 250
      IF ( XCONN(ICOL5,ICOL6).EQ.CCTMX(3) ) GO TO 250
      IF ( XCONN(ICOL5,ICOL6).EQ.CCTMX(4) ) GO TO 250
      IF ( ICOL6.EQ.ICOL1.OR.ICOL6.EQ.ICOL2.OR.
     *     ICOL6.EQ.ICOL3.OR.ICOL6.EQ.ICOL4 ) GO TO 250
      NCTMX(6,1) = ICOL5
      NCTMX(6,2) = ICOL6
      CCTMX(6) = XCONN(ICOL5,ICOL6)
      CCTMY(6) = YCONN(ICOL5,ICOL6)
      IF ( CCTMX(6).GE.CCTMX(5).AND.CCTMY(6).LE.CCTMY(5) )
     *     GO TO 250
C
      IF ( ICOL6.EQ.1.AND.CCTMX(6).NE.0. ) THEN
         IF ( XCONN(ICOL5,1).GT.CCTMX(1) ) THEN
              KOPTN = 0
              CALL KINET ( 6, CCTMX, CCTMY, KOPTN, YPERT )
              IF ( KOPTN.EQ.1 ) GO TO 250
              NPATH = NPATH + 1
              TPATH = TPATH + 1.
              NITER = -6
              CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ,
     *                     PROPT, SVALU, TVALU )
              IF ( JFLAG.EQ.1 )
     *           CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         ENDIF
         GO TO 250
      ENDIF
C
      NFLAG = 0
      IF ( CCTMY(6).LT.CCTMY(5) ) CALL KINES ( 6, CCTMX, CCTMY, NFLAG )
      IF ( NFLAG.EQ.1 ) GO TO 250
C
      IF ( ICOL6.EQ.NTRCE.AND.XCONN(ICOL5,NTRCE).NE.0. ) THEN
         KOPTN = 0
         CALL KINET ( 6, CCTMX, CCTMY, KOPTN, YPERT )
         IF ( KOPTN.EQ.1 ) GO TO 250
         NPATH = NPATH + 1
         NITER = 6
         CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ, PROPT,
     *                SVALU, TVALU )
         IF ( JFLAG.FQ.1 )
     *      CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX  YPERT )
         GO TO 250
      ENDIF
C
```

```
C     7TH
C
      DO 240 ICOL7 = 1, NTRCE
      IF ( XCONN(ICOL6,ICOL7).EQ.0.        ) GO TO 240
      IF ( XCONN(ICOL6,ICOL7).EQ.CCTMX(1) ) GO TO 240
      IF ( XCONN(ICOL6,ICOL7).EQ.CCTMX(2) ) GO TO 240
      IF ( XCONN(ICOL6,ICOL7).EQ.CCTMX(3) ) GO TO 240
      IF ( XCONN(ICOL6,ICOL7).EQ.CCTMX(4) ) GO TO 240
      IF ( XCONN(ICOL6,ICOL7).EQ.CCTMX(5) ) GO TO 240
      IF ( ICOL7.EQ.ICOL1.OR.ICOL7.EQ.ICOL2.OR.
     *     ICOL7.EQ.ICOL3.OR.ICOL7.EQ.ICOL4.OR.
     *     ICOL7.EQ.ICOL5 ) GO TO 240
      NCTMX(7,1) = ICOL6
      NCTMX(7,2) = ICOL7
      CCTMX(7) = XCONN(ICOL6,ICOL7)
      CCTMY(7) = YCONN(ICOL6,ICOL7)
      IF ( CCTMX(7).GE.CCTMX(6).AND.CCTMY(7).LE.CCTMY(6) )
     *     GO TO 240
C
      IF ( ICOL7.EQ.1.AND.CCTMX(7).NE.0. ) THEN
         IF ( XCONN(ICOL6,1).GT.CCTMX(1) ) THEN
            KOPTN = 0
            CALL KINET ( 7, CCTMX, CCTMY, KOPTN, YPERT )
            IF ( KOPTN.EQ.1 ) GO TO 240
            NPATH = NPATH + 1
            TPATH = TPATH + 1.
            NITER = -7
            CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ.
     *                   PROPT, SVALU, TVALU )
            IF ( JFLAG.EQ.1 )
     *         CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         ENDIF
         GO TO 240
      ENDIF
C
      NFLAG = 0
      IF ( CCTMY(7).LT.CCTMY(6) ) CALL KINES ( 7, CCTMX, CCTMY, NFLAG )
      IF ( NFLAG.EQ.1 ) GO TO 240
C
      IF ( ICOL7.EQ.NTRCE.AND.XCONN(ICOL6,NTRCE).NE.0. ) THEN
         KOPTN = 0
         CALL KINET ( 7, CCTMX, CCTMY, KOPTN, YPERT )
         IF ( KOPTN.EQ.1 ) GO TO 240
         NPATH = NPATH + 1
         NITER = 7
         CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ, PROPT,
     *                SVALU, TVALU )
         IF ( JFLAG.EQ.1 )
     *      CALL PORTH ( NITER,CCTMX, CCTMY, NCTMX, YPERT )
         GO TO 240
      ENDIF
C
C     8TH
C
      DO 230 ICOL8 = 1, NTRCE
      IF ( XCONN(ICOL7,ICOL8).EQ.0.        ) GO TO 230
      IF ( XCONN(ICOL7,ICOL8).EQ.CCTMX(1) ) GO TO 230
      IF ( XCONN(ICOL7,ICOL8).EQ.CCTMX(2) ) GO TO 230
      IF ( XCONN(ICOL7,ICOL8).EQ.CCTMX(3) ) GO TO 230
      IF ( XCONN(ICOL7,ICOL8).EQ.CCTMX(4) ) GO TO 230
      IF ( XCONN(ICOL7,ICOL8).EQ.CCTMX(5) ) GO TO 230
      IF ( XCONN(ICOL7,ICOL8).EQ.CCTMX(6) ) GO TO 230
      IF ( ICOL8.EQ.ICOL1.OR.ICOL8.EQ.ICOL2.OR.
     *     ICOL8.EQ.ICOL3.OR.ICOL8.EQ.ICOL4.OR.
     *     ICOL8.EQ.ICOL5.OR.ICOL8.EQ.ICOL6    ) GO TO 230
      NCTMX(8,1) = ICOL7
      NCTMX(8,2) = ICOL8
      CCTMX(8) = XCONN(ICOL7,ICOL8)
      CCTMY(8) = YCONN(ICOL7,ICOL8)
      IF ( CCTMX(8).GE.CCTMX(7).AND.CCTMY(8).LE.CCTMY(7) )
     *     GO TO 230
C
      IF ( ICOL8.EQ.1.AND.CCTMX(8).NE.0. ) THEN
         IF ( XCONN(ICOL7,1).GT.CCTMX(1) ) THEN
            KOPTN = 0
            CALL KINET ( 8, CCTMX, CCTMY, KOPTN, YPERT )
            IF ( KOPTN.EQ.1 ) GO TO 230
            NPATH = NPATH + 1
            TPATH = TPATH + 1.
            NITER = -8
            CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ,
     *                   PROPT, SVALU, TVALU )
            IF ( JFLAG.EQ.1 )
     *         CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         ENDIF
         GO TO 230
      ENDIF
C
      NFLAG = 0
      IF ( CCTMY(8).LT.CCTMY(7) ) CALL KINES ( 8, CCTMX, CCTMY, NFLAG )
      IF ( NFLAG.EQ.1 ) GO TO 230
C
      IF ( ICOL8.EQ.NTRCE.AND.XCONN(ICOL7,NTRCE).NE.0. ) THEN
         KOPTN = 0
         CALL KINET ( 8, CCTMX, CCTMY, KOPTN, YPERT )
         IF ( KOPTN.EQ.1 ) GO TO 230
         NPATH = NPATH + 1
         NITER = 8
         CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ, PROPT,
     *                SVALU, TVALU )
         IF ( JFLAG.EQ.1 )
     *      CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         GO TO 230
      ENDIF
C
C     9TH
C
      DO 220 ICOL9 = 1, NTRCE
      IF ( XCONN(ICOL8,ICOL9).EQ.0.  ) GO TO 220
      DO 180 JITER = 1, 7
```

```
         IF ( XCONN(ICOL8,ICOL9).EQ.CCTMX(JITER) ) GO TO 220
 180  CONTINUE
      IF ( ICOL9.EQ.ICOL1.OR.ICOL9.EQ.ICOL2.OR.ICOL9.EQ.ICOL3.OR.
     *     ICOL9.EQ.ICOL4.OR.ICOL9.EQ.ICOL5.OR.ICOL9.EQ.ICOL6.OR.
     *     ICOL9.EQ.ICOL7 ) GO TO 220
      NCTMX(9,1) = ICOL8
      NCTMX(9,2) = ICOL9
      CCTMX(9) = XCONN(ICOL8,ICOL9)
      CCTMY(9) = YCONN(ICOL8,ICOL9)
      IF ( CCTMX(9).GE.CCTMX(8).AND.CCTMY(9).LE.CCTMY(8) )
     *     GO TO 220
C
      IF ( ICOL9.EQ.1.AND.CCTMX(9).NE.0. ) THEN
         IF ( XCONN(ICOL8,1).GT.CCTMX(1) ) THEN
            KOPTN = 0
            CALL KINET ( 9, CCTMX, CCTMY, KOPTN, YPERT )
            IF ( KOPTN.EQ.1 ) GO TO 220
            NPATH = NPATH + 1
            TPATH = TPATH + 1.
            NITER = -9
            CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ,
     *                   PROPT, SVALU, TVALU )
            IF ( JFLAG.EQ.1 )
     *         CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         ENDIF
         GO TO 220
      ENDIF
C
      NFLAG = 0
      IF ( CCTMY(9).LT.CCTMY(8) ) CALL KINES ( 9, CCTMX, CCTMY, NFLAG )
      IF ( NFLAG.EQ.1 ) GO TO 220
C
      IF ( ICOL9.EQ.NTRCE.AND.XCONN(ICOL8,NTRCE).NE.0. ) THEN
         KOPTN = 0
         CALL KINET ( 9, CCTMX, CCTMY, KOPTN, YPERT )
         IF ( KOPTN.EQ.1 ) GO TO 220
         NPATH = NPATH + 1
         NITER = 9
         CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ, PROPT,
     *                SVALU, TVALU )
         IF ( JFLAG.EQ.1 )
     *      CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         GO TO 220
      ENDIF
C
C     10TH
C
      DO 210 ICOL0 = 1, NTRCE
      IF ( XCONN(ICOL9,ICOL0).EQ.0. ) GO TO 210
      DO 190 JITER = 1, 8
         IF ( XCONN(ICOL9,ICOL0).EQ.CCTMX(JITER) ) GO TO 210
 190  CONTINUE
      IF ( ICOL0.EQ.ICOL1.OR.ICOL0.EQ.ICOL2.OR.ICOL0.EQ.ICOL3.OR.
     *     ICOL0.EQ.ICOL4.OR.ICOL0.EQ.ICOL5.OR.ICOL0.EQ.ICOL6.OR.
     *     ICOL0.EQ.ICOL7.OR.ICOL0.EQ.ICOL8 ) GO TO 210
```

```
      NCTMX(10,1) = ICOL9
      NCTMX(10,2) = ICOL0
      CCTMX(10) = XCONN(ICOL9,ICOL0)
      CCTMY(10) = YCONN(ICOL9,ICOL0)
      IF ( CCTMX(10).GE.CCTMX(9).AND.CCTMY(10).LE.CCTMY(9) )
     *     GO TO 210
C
      IF ( ICOL0.EQ.1.AND.CCTMX(10).NE.0. ) THEN
         IF ( XCONN(ICOL9,1).GT.CCTMX(1) ) THEN
            KOPTN = 0
            CALL KINET ( 10, CCTMX, CCTMY, KOPTN, YPERT )
            IF ( KOPTN.EQ.1 ) GO TO 210
            NPATH = NPATH + 1
            TPATH = TPATH + 1.
            NITER = -10
            CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ,
     *                 PROPT, SVALU, TVALU )
            IF ( JFLAG.EQ.1 )
     *         CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         ENDIF
         GO TO 210
      ENDIF
C
      IF ( ICOL0.EQ.NTRCE.AND.XCONN(ICOL9,NTRCE).NE.0. ) THEN
         KOPTN = 0
         CALL KINET ( 10, CCTMX, CCTMY, KOPTN, YPERT )
         IF ( KOPTN.EQ.1 ) GO TO 210
         NPATH = NPATH + 1
         NITER = 10
         CALL FMODE ( NITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ, PROPT,
     *                SVALU, TVALU )
         IF ( JFLAG.EQ.1 )
     *      CALL PORTH ( NITER, CCTMX, CCTMY, NCTMX, YPERT )
         GO TO 210
      ENDIF
      LCOUN = LCOUN + 1
C
 210  CONTINUE
 220  CONTINUE
 230  CONTINUE
 240  CONTINUE
 250  CONTINUE
 260  CONTINUE
 270  CONTINUE
 280  CONTINUE
 290  CONTINUE
 300  CONTINUE
      IF ( LCOUN.GT.0 ) WRITE(6,950) LCOUN
C
C     STORE THE TOTAL NUMBER OF EFFECTIVE JOINT PATHS
C
      MPATH = TPATH
      WRITE(6,920) NPATH, MPATH
      IF ( MOPTN.GE.5 ) WRITE(7,940) NPATH, MPATH
C
```

```
C     TOTAL JOINT PATHES WHICH ARE MECHANICALLY UNSTABLE : NMODE
C
      WRITE(6,930) NMODE
      WRITE(9,925) ICASE, IOSIM, NPATH, NMODE
      WRITE(7,945) NMODE
      IF ( NMODE.GE.1 ) THEN
         WRITE(6,990) SFMIN
         CALL PORTH ( NCRIT, CRITX, CRITY, NCTMX, YCRIT )
         WRITE(8,965) SFMIN
      ENDIF
C
C     STORE THE STATISTICAL INFORMATION INTO TAPE 9
C
      IF ( NSTAT.GT.0.AND.MOPTN.GE.5 ) THEN
         DO 500 ISTEP = 1, NSTEP
         VALUE = VARMN + (ISTEP-1) * DSTEP
         WRITE(9,955) VALUE, SVALU(IOSIM,ISTEP), TVALU(IOSIM,ISTEP)
 500     CONTINUE
      ENDIF
      IF ( IOSIM.EQ.NOSIM.AND.NSTAT.GT.0 ) THEN
         IF ( ICASE.EQ.NCASE ) WRITE(6,975)
         DO 600 ISTEP = 1, NSTEP
         SVALE = 0.
         TVALE = 0.
         DO 550 JOSIM = 1, NOSIM
         SVALE = SVALE + SVALU(JOSIM,ISTEP)
         TVALE = TVALE + TVALU(JOSIM,ISTEP)
 550     CONTINUE
         SVALU(MOSIM,ISTEP) = SVALE
         TVALU(MOSIM,ISTEP) = TVALE
         VALUE = VARMN + ( ISTEP - 1 ) * DSTEP
C
         CASEO(ICASE,ISTEP) = SVALE
         CASET(ICASE,ISTEP) = TVALE
         IF ( ICASE.EQ.NCASE )
     *    WRITE(6,956) VALUE, (CASEO(JCASE,ISTEP), CASET(JCASE,ISTEP),
     *        JCASE=1,NCASE)
 600     CONTINUE
      ENDIF
C
C     STORE THE FINAL STABILITY ANALYSIS ONTO FILE9
C
      IF ( ICASE.EQ.NCASE.AND.IOSIM.EQ.NOSIM.AND.
     *     NSTAT.NE.O ) THEN
         WRITE(9,940) NCASE, NSTEP, NOSIM, NSTAT
         WRITE(9,935)
         WRITE(6,980)
         DO 710 ISTEP = 1, NSTEP
         VALUE = VARMN + (ISTEP-1) * DSTEP
         DO 715 JCASE = 1, NCASE
         IF ( CASEO(JCASE,ISTEP).EQ.0. ) THEN
            PROBY(JCASE) = 0.
         ELSE
            PROBY(JCASE) = CASET(JCASE,ISTEP) / CASEO(JCASE,ISTEP)
         ENDIF
 715     CONTINUE
         WRITE(6,955) VALUE, (PROBY(JCASE),JCASE=1,NCASE)
         WRITE(9,955) VALUE, (PROBY(JCASE),JCASE=1,NCASE)
 710     CONTINUE
C
C     CALCULATE THE PROBABILITY OF FAILURE OF EACH VOLUME OR DIP
C
         WRITE(9,935)
         WRITE(6,985)
         DO 730 JSTEP = 1, NSTEP
         TPRBY(JSTEP) = 0.
         DO 720 JCASE = 1, NCASE
         TPRBY(JSTEP) = TPRBY(JSTEP) + CASET(JCASE,JSTEP)
 720     CONTINUE
 730     CONTINUE
         DO 750 JSTEP = 1, NSTEP
         VALUE = VARMN + (JSTEP-1) * DSTEP
         DO 740 JCASE = 1, NCASE
         IF ( TPRBY(JSTEP).EQ.0. ) THEN
            PROBY(JCASE) = 0.
         ELSE
            PROBY(JCASE) = CASET(JCASE,JSTEP) / TPRBY(JSTEP)
         ENDIF
 740     CONTINUE
         WRITE(6,960) VALUE, (PROBY(KCASE),KCASE=1,NCASE)
         WRITE(9,960) VALUE, (PROBY(KCASE),KCASE=1,NCASE)
 750     CONTINUE
C
C     STORE THE INFORMATION ONTO TAPE 9
C
         WRITE(9,935)
         DO 760 KSTEP = 1, NSTEP
         VALUE = VARMN + ( KSTEP-1 ) * DSTEP
         WRITE(9,956) VALUE, (CASEO(JCASE,KSTEP), CASET(JCASE,KSTEP),
     *                JCASE=1,NCASE)
 760     CONTINUE
      ENDIF
C
 920  FORMAT(//, 5X, 'TOTAL NO. OF REAL EFFECTIVE PATHS = ', I5,
     *        /, 5X, 'TOTAL NO. OF FACE TO FACE PATHS   = ', I5 )
 925  FORMAT(4I5)
 930  FORMAT(//, 5X, 'TOTAL NO. OF UNSTABLE JOINT PATHS = ', I5 )
 935  FORMAT(/)
 940  FORMAT(4I5)
 945  FORMAT(I5)
 950  FORMAT(//, 5X, '*** 10 CONSECUTIVE SEARCHING ORDER IS NOT ',
     *               'SUFFICIENT ***',
     *        /, 7X, 'TOTAL PATHS OVER 10 JOINTS ARE = ', I5      )
 955  FORMAT(6X, F7.2, 5(F7.2,3X))
 956  FORMAT(6X, F7.2, 5(F7.2,3X,F7.2,3X))
 960  FORMAT(6F10.3)
 965  FORMAT(F10.3)
 970  FORMAT(//, 5X, '*** WARNING ***',
     *        /, 7X, 'CALCULATED INTERSECTION POINTS = ', I5,
     *        /, 7X, 'STORED     INTERSECTION POINTS = ', I5 )
 975  FORMAT(//, 5X, 'TOTAL NUMBER OF JOINT PATHS',
     *        /, 4X, 'VALUE     TOTAL PATHS  UNSTABLE PATHS',
     *        /)
 980  FORMAT(//, 5X, 'Pf AT EACH VOLUME CATEGORY',
     *        /, 3X, 'INCREMENT',3X,'Pf FOR EACH CASE',/)
 985  FORMAT(//, 5X, 'Pf OF EACH VOLUME AMONG UNSTABLE PATHS',
     *        /, 4X, 'INCREMENT', 6X, 'Pf FOR EACH CASE', /)
 990  FORMAT(//, 5X, 'MIN. FACTOR OF SAFETY = ',        F10.3,
     *        /, 7X, '( RESISTING FORCE / DRIVING FORCE )'   )
C
      RETURN
      END
CCC
C
      SUBROUTINE PORTH ( LITER, CCTMX, CCTMY, NCTMX, YPERT )
C
C     SUBROUTINE PORTH STORES THE JOINT PATH COORDINATES
C     IN FILE 8
C
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *              NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      DIMENSION CCTMX(10), CCTMY(10), NCTMX(10,2)
C
      ANRAD = 3.1415926 / 180.
      XPERT = -0.5
      XRANG = HEIGT / TAN(ANGLE*ANRAD)
      NITER = ABS(LITER)
C
      IF ( MOPTN.GE.5 ) THEN
         WRITE(8,910) NITER
         DO 10 IITER = 1,NITER
         WRITE(8,920) CCTMX(IITER), CCTMY(IITER)
 10      CONTINUE
      ENDIF
      IF ( MOPTN.GE.5 ) THEN
         IF ( LITER.LT.0 ) THEN
            KITER = NITER + 1
            WRITE(8,910) KITER
            DO 20 IITER = 1, NITER
               CCTMX1 = CCTMX(IITER) + XPERT
               CCTMY1 = CCTMY(IITER) + YPERT
               WRITE(8,920) CCTMX1, CCTMY1
 20         CONTINUE
            CCTMX1 = CCTMX(1) + XPERT
            CCTMY1 = CCTMY(1) + YPERT
            WRITE(8,920) CCTMX1, CCTMY1
         ELSE
            KITER = NITER + 2
            WRITE(8,910) KITER
            DO 30 IITER = 1, NITER
               CCTMX1 = CCTMX(IITER) + XPERT
               CCTMY1 = CCTMY(IITER) + YPERT
               WRITE(8,920) CCTMX1, CCTMY1
 30         CONTINUE
            CCTMX1 = XRANG + XPERT
            CCTMY1 = HEIGT + YPERT
            WRITE(8,920) CCTMX1, CCTMY1
            CCTMX1 = CCTMX(1) + XPERT
            CCTMY1 = CCTMY(1) + YPERT
            WRITE(8,920) CCTMX1, CCTMY1
         ENDIF
      ENDIF
C
 910  FORMAT(I5)
 920  FORMAT(2F10.3)
C
      RETURN
      END
CCC
C
      SUBROUTINE KINES ( NITER, CCTMX, CCTMY, NFLAG )
C
C     SUBROUTINE LINES TESTS WHETHER THE ASSUMED JOINT PATH IS
C     KINEMATICALLY ADMISSIBLE OR NOT.
C
      DIMENSION CCTMX(10), CCTMY(10)
C
      NFLAG = 0
C
      ITER1 = NITER - 1
      ITER2 = NITER - 2
      XCOR1 = CCTMX(ITER1)
      YCOR1 = CCTMY(ITER1)
      XCOR2 = CCTMX(ITER2)
      YCOR2 = CCTMY(ITER2)
      XCOOR = CCTMX(NITER)
      YCOOR = CCTMY(NITER)
C
      IF ( YCOR1.EQ.YCOR2 ) THEN
         IF ( YCOOR.LT.YCOR1 ) THEN
            NFLAG = 1
            GO TO 50
         ENDIF
      ELSE
         IF ( XCOR1.EQ.XCOR2 ) THEN
            IF ( XCOOR.GT.XCOR1 ) THEN
               NFLAG = 1
               GO TO 50
            ENDIF
         ENDIF
         SLOPE = ( YCOR2 - YCOR1 ) / ( XCOR2 - XCOR1 )
         YCOR3 = ( YCOR2-YCOR1 ) * ( XCOOR-XCOR1 ) / ( XCOR2-XCOR1 )
     *         + YCOR1
         IF ( SLOPE.GT.0..AND.YCOR3.GT.YCOOR ) NFLAG = 1
         IF ( SLOPE.LT.0..AND.YCOR3.LT.YCOOR ) NFLAG = 1
      ENDIF
C
 50   CONTINUE
C
      RETURN
      END
CCC
C
      SUBROUTINE KINET ( MITER, CCTMX, CCTMY, KOPTN, YPERT )
C
C     SUBROUTINE KINET CHECKS THE ADMISSIBILITY OF
C     AN ASSUMED JOINT PATH USING MINIMUM DIP AND
C     PERTURBATION BEHAVIOR
C
      DIMENSION CCTMX(10), CCTMY(10)
      DIMENSION SLOPE(10)
C
      PIRAD = 3.1415926
      XPERT = -0.5
      KOPTN = 0
C
C     FIND THE MINIMUM DIP
C
      MPITR = MITER - 1
      IF ( MITER.EQ.2 ) GO TO 30
      DO 10 IITER = 1, MPITR
      JITER = IITER + 1
      XCOR1 = CCTMX(IITER)
      YCOR1 = CCTMY(IITER)
      XCOR2 = CCTMX(JITER)
      YCOR2 = CCTMY(JITER)
      IF ( XCOR1.EQ.XCOR2 ) XCOR2 = XCOR2 + 1.E-10
      DIPAN = ( YCOR2 - YCOR1 ) / ( XCOR2 - XCOR1 )
      SLOPE(IITER) = ATAN(DIPAN)
      IF ( SLOPE(IITER).LT.0. ) SLOPE(IITER) = SLOPE(IITER) + PIRAD
      IF ( XCOR2.LT.XCOR1.AND.YCOR2.LT.YCOR1 )
     *     SLOPE(IITER) = SLOPE(IITER) + PIRAD
C
C     DETERMINE MIN. DIP ANGLE ( RAD. )
C
      IF ( IITER.EQ.1 ) THEN
         SLMIN = SLOPE(1)
         INDSP = 1
      ELSE
         IF ( SLOPE(IITER).GT.0..AND.SLOPE(IITER).LT.SLMIN ) THEN
            SLMIN = SLOPE(IITER)
            INDSP = IITER
         ENDIF
      ENDIF
C
 10   CONTINUE
C
C     CHECK THE MOBILITY OF A JOINT PATH
C
      DO 20 IITER = 1, MPITR
      IF ( SLOPE(IITER).LT.PIRAD ) GO TO 20
      SLTRY = SLOPE(IITER) - PIRAD
      IF ( SLTRY.GT.SLMIN ) THEN
         KOPTN = 1
         RETURN
      ENDIF
 20   CONTINUE
C
C     FIND PURTURBED Y-COORDINATE
C
 30   CONTINUE
      IF ( MITER.EQ.2 ) THEN
         YPERT = ( CCTMY(2)-CCTMY(1) ) / ( CCTMX(2)-CCTMX(1) )
     *         * ( XPERT )
      ELSE
         YPERT = SLMIN * ( XPERT )
      ENDIF
C
      RETURN
      END
CCC
C
      SUBROUTINE FMODE ( LITER, CCTMX, CCTMY, JFLAG, NMTRL, NOEFJ,
     *                   PROPT, SVALU, TVALU )
C
C     SUBROUTINE FMODE EVALUATES THE MECHANICAL FAILURE MODE
C     OF AN ASSUMED JOINT PATH USING COULOMB'S SHEAR FAILURE
C     CRITERION
C
      COMMON/CONTRO/NPOIN, NTRCE, NOSIM, NDOFN, MOPTN, IOSIM, NSYST,
     *              NOSET, ANGLE, HEIGT, RANGE, SFMAX, WEIGT, NCASE
      COMMON/CONTOL/NMODE, SFMIN, MOSIM
      COMMON/CNTROL/CRITX(10), CRITY(10), NCRIT, YCRIT
      COMMON/CONROL/NCTMX(10,2)
      COMMON/STATIC/NSTAT, VARMN, DSTEP, NSTEP, ICASE, NREG1, NREG2
      DIMENSION NMTRL(NOEFJ), PROPT(NOSET,7)
      DIMENSION CCTMX(10), CCTMY(10), TLENG(10), DIPTH(10)
      DIMENSION SVALU(MOSIM,NSTEP), TVALU(MOSIM,NSTEP)
C
      PIRAD = 3.1415926
      ANRAD = PIRAD / 180.
      XRANG = HEIGT / TAN(ANGLE*ANRAD)
      SLOPF = TAN(ANGLE*ANRAD)
      JFLAG = 0
C
C     CALCULATES THE WHOLE AREA
C
      KITER = 0
      AREAT = 0.
      NITER = ABS(LITER) - 1
      DO 40 IITER = 1, NITER
      DIPTH(IITER) = 0.
      JITER = IITER + 1
      XCOR1 = CCTMX(IITER)
      XCOR2 = CCTMX(JITER)
      YCOR1 = CCTMY(IITER)
      YCOR2 = CCTMY(JITER)
      XDIFF = XCOR2 - XCOR1
      YDIFF = YCOR2 - YCOR1
      TLENG(IITER) = SQRT( XDIFF**2 + YDIFF**2 )
      IF ( XDIFF.EQ.0. ) GO TO 40
      SLOPE = YDIFF / XDIFF
      IF ( XCOR2.LT.XCOR1.AND.YCOR2.LE.YCOR1 ) SLOPE = -SLOPE
      IF ( XCOR2.GT.XCOR1.AND.YCOR2.LT.YCOR1 ) THEN
         WRITE(6,910)
         RETURN
      ENDIF
C
      IF ( SLOPE.LT.0. ) GO TO 20
      KITER  = KITER + 1
      IF ( KITER.EQ.1 ) THEN
         SLMIN = SLOPE
         MINJT = 1
      ELSE
         IF ( SLOPE.LT.SLMIN ) THEN
            SLMIN = SLOPE
            MINJT = IITER
         ENDIF
      ENDIF
      DIPTH(IITER) = SLOPE
      JOINT = NCTMX(MINJT,2) - 1
 20   CONTINUE
C
      IF ( XCOR1.GE.XRANG.AND.XCOR2.GE.XRANG ) GO TO 30
      IF ( XCOR1.LT.XRANG.AND.XCOR2.LT.XRANG ) THEN
         AREA1 = 0.5 * ( 2.* HEIGT - YCOR1 - YCOR2 ) * ABS(XDIFF)
         YCOR1 = SLOPF * XCOR1
         YCOR2 = SLOPF * XCOR2
         AREA2 = 0.5 * ( 2.* HEIGT - YCOR1 - YCOR2 ) * ABS(XDIFF)
      ELSE
         AREA1 = 0.5 * ( 2.* HEIGT - YCOR1 - YCOR2 ) * ABS(XDIFF)
         IF ( SLOPE.GE.0. ) THEN
            XCOOR = XCOR1
         ELSE
            XCOOR = XCOR2
         ENDIF
         YCOOR = SLOPF * XCOOR
         AREA2 = 0.5 * ( XRANG - AMIN1(XCOR1,XCOR2) ) *
     *                 ( HEIGT - YCOOR )
      ENDIF
      AREA3 = AREA1 - AREA2
      IF ( AREA3.LE.0. ) WRITE(6,900)
      IF ( SLOPE.LT.0. ) AREA3 = -AREA3
      AREAT = AREAT + AREA3
      GO TO 40
C
 30   CONTINUE
C
      AREA1 = 0.5 * ( 2.* HEIGT - YCOR1 - YCOR2 ) * ABS(XDIFF)
      IF ( SLOPE.LT.0. ) AREA1 = -AREA1
      AREAT = AREAT + AREA1

 40   CONTINUE
      IF ( AREAT.LE.0. )  WRITE(6,900)
C
```

```
C     USING COULOMB'S LAW ON A MIN. DIP JOINT FACE, CHECK THE
C     POSSIBILITY OF SLIDING OF ROCK BLOCK
C
      TOWGT = WEIGT * AREAT
      DIPMN = ATAN(SLMIN)
      DRIVE = TOWGT * SIN(DIPMN)
C
      TOTCH = 0.
      TENCU = 0.
      DO 50 IITER = 1, NITER
C
C     INCLUDE THE POSSIBILITY OF THE PARALLEL DIP ANGLE
C
      JTSPT = NCTMX(IITER,2) - 1
      NOMAT = NMTRL(JTSPT)
      IF ( DIPTH(IITER).EQ.SLMIN ) THEN
         COHJT = PROPT(NOMAT,5) * TLENG(IITER)
         TOTCH = TOTCH + COHJT
      ELSE
         TENJT = PROPT(NOMAT,7) * TLENG(IITER)
         TENCU = TENCU + TENJT
      ENDIF
C
 50   CONTINUE
      NOMAT = NMTRL(JOINT)
      FRIJT = PROPT(NOMAT,6)
      RESIT = TOTCH + TOWGT * COS(DIPMN) * TAN(FRIJT) + TENCU
C
      SAFEF = RESIT / DRIVE
C
C     STATISTICAL ANALYSIS FOR VOLUME ( NSTAT = 1 )
C
      IF ( NSTAT.EQ.2  ) GO TO 200
C
      ICTGY = ( AREAT - VARMN ) / DSTEP + 2
      IF ( AREAT.LT.VARMN ) ICTGY = 1
      IF ( ICTGY.GT.NSTEP ) ICTGY = NSTEP
      SVALU(IOSIM,ICTGY) = SVALU(IOSIM,ICTGY) + 1.
C
      IF ( NSTAT.NE.1.AND.ICASE.LE.NREG2.AND.IOSIM.LE.NREG1 ) THEN
         WRITE(10,940) ICASE, IOSIM, AREAT, DIPMN, TOTCH, TENCU,
     *          RESIT, DRIVE
      ENDIF
C
 200  CONTINUE
C
C     STATISTICAL ANALYSIS FOR MIN. DIP ANGLE ( NSTAT = 2 )
C
      IF ( NSTAT.NE.2 ) GO TO 300
      DIPDG = DIPMN / ANRAD
      ICTGY = ( DIPDG - VARMN ) / DSTEP + 2
      IF ( DIPDG.LT.VARMN ) ICTGY = 1
      IF ( ICTGY.GT.NSTEP ) ICTGY = NSTEP
      SVALU(IOSIM,ICTGY) = SVALU(IOSIM,ICTGY) + 1.
C
```

```
  300 CONTINUE
      IF ( SAFEF.GT.SFMAX ) RETURN
C
C     STORE THE UNSTABLE JOINT PATH INTO TVALU
C
      IF ( NSTAT.EQ.1.AND.AREAT.LT.VARMN ) GO TO 400
      IF ( NSTAT.EQ.2.AND.DIPDG.LT.VARMN ) GO TO 400
      IF ( NSTAT.GT.0.AND.ICTGY.LE.NSTEP ) THEN
         TVALU(IOSIM,ICTGY) = TVALU(IOSIM,ICTGY) + 1.
      ENDIF
C
  400 CONTINUE
      JFLAG = 1
      NMODE = NMODE + 1
      IF ( NMODE.EQ.1 ) THEN
         SFMIN = SAFEF
      ELSE
         IF ( SAFEF.LT.SFMIN ) SFMIN = SAFEF
      ENDIF
C
C     STORE THE COORDINATES OF THE CRITICAL JOINT PATH FOR PLOTTING
C
      IF ( SFMIN.EQ.SAFEF ) THEN
         NITER = ABS(LITER)
         DO 60 IITER = 1, NITER
            CRITX(IITER) = CCTMX(IITER)
            CRITY(IITER) = CCTMY(IITER)
  60     CONTINUE
            NCRIT = LITER
            XPERT = -0.5
            YCRIT = SLMIN * XPERT
      ENDIF
C
  900 FORMAT(//,5X,'*** NEGATIVE AREA ***' )
  910 FORMAT(//,5X,'*** ENCOUNTERED A KINEMATICALLY INADMISSIBLE PATH')
  940 FORMAT(2X,2I3,2X,6(F10.3,1X))
C
      RETURN
      END
```

References

[1] G. Augusti, A. Baratta and F. Casciati.
Probabilistic Methods in Structural Engineering.
Chapman & Hall, 1984.

[2] G.B. Baecher, N.A. Lanney and H.H. Einstein.
Statistical Description of Rock Properties and Sampling.
Proc. 18th U.S. Symp. Rock Mech. :5C1-5C8, 1977.

[3] G.A. Barnard.
contribution to the Discussion of Prof. Bartlett's Paper.
J. R. Statist. Soc., B. 25:294, 1963.

[4] M.S. Bartlett.
Spectral Analysis of Two-dimensional Point Processes.
Biometrika 51:299-311, 1964.

[5] C.C. Barton and E. Larson.
Fractal Geometry of Two-dimensional Fracture Networks at Yucca Mountain, Southwest Nevada.
Fundamentals of Rock Joints : Proc. of Int. Symp. on Fundamentals of Rock Joints, Ed. O. Stephannson :77-84, 1985.

[6] J.E. Besag.
Contribution to the Discussion of Ripley's Paper.
J. R. Statist. Soc. B. 39:193-195, 1977.

[7] J.E. Besag and J.T. Gleaves.
On the Detection of Spatial Pattem in Plant Communities.
Bull. Int. Statist. Inst. 45:153-158, 1973.

[8] J.E. Besag and P.J. Diggle.
Simple Monte Carlo Tests for Spatial Pattem.
Appl. Statist. 26:327-333, 1977.

[9] D. Brown and P. Rothery.
Randomness and Local Regularity of Points in a Plane.
Biometrika 65:115-122, 1978.

[10] K. Byth and B.D. Ripley.
On Sampling Spatial Pattems by Distance Methods.
Biometrics 36:279-284, 1980.

[11] J.P. Chiles.
Fractal and Geostatistical Methods for Modeling of a Fractured Network.
Math. Geol. 20:631-654, 1988.

[12] P.J. Clark and F.C. Evans.
Distance to Nearest Neighbor as a Measure of Spatial Relationships in Populations.
Ecology 35:445-453, 1954.

[13] F. Conrad and C. Jacquin.
Representation of a Two-dimensional Fracture Network by a Probabilistic Model : Application to Calculation of the Geometric Magnitudes of Matrix Blocks.
Technical Report UCRL-TRANS-10814, U.C. Lawrence Livermore Lab., 1975.

[14] R. Cowan.
Homogeneous Line-Segment Processes.
Math. Proc. Camb. Phil. Soc. 86:481-489, 1979.

[15] T.F. Cox and T. Lewis.
A Conditioned Distance Ratio Method for Analyzing Spatial Patterns.
Biometrika 63:483-491, 1976.

[16] W.S. Dershowitz.
Rock Joint Systems.
PhD thesis, M.I.T., 1985.

[17] W.S. Dershowitz.
Personal Communication.
1988.

[18] W.S. Dershowitz and H.H. Einstein.
Characterizing Rock Joint Geometry with Joint System Models.
Rock Mech & Rock Eng. 21:21-51, 1988.

[19] P.J. Diggle.
The Detection of Random Heterogeneity in Plant Populations.
Biometrics 33:390-394, 1977.

[20] P.J. Diggle.
Some Graphical Methods in the Analysis of Spatial Point Patterns.
In V. Barnett (editor), *Interpreting Multivariate Data*, pages 55-73. 1981.

[21] P.J. Diggle.
Statistical Analysis of Spatial Point Patterns.
Academic Press, 1983.

[22] P.J. Diggle, J. Besag and J.T. Gleaves.
Statistical Analysis of Spatial Point Patterns by Means of Distance Methods.
Biometrics 32:659-667, 1976.

[23] L.L. Eberhardt.
Some Developments in Distance Sampling.
Biometrics 23:207-216, 1967.

[24] H.H. Einstein, et al.
Risk Analysis for Rock Slopes in Open Pit Mines.
Technical Report J 027 5015, U.S. Bureau of Mines, 1979.

[25] H.H. Einstein and G.B. Baecher.
Probabilistic and Statistical Methods in Engineering Geology. Part I : Exploration.
Rock Mech & Rock Eng. 16:39-72, 1983.

[26] Einstein, H. H., et al.
The Effect of Discontinuity Persistence on Rock Slope Stability.
Int. J. Rock Mech. Min. Sci. :227-236, 1983.

[27] V.A. Epanechnikov.
Non-parametric Estimation of a Multivariate Probability Density.
Theor. Prob. App. 14:153-158, 1969.

[28] T. Fiksel.
Edge-corrected Density Estimators for Point Processes.
Statistics 19:67-75, 1988.

[29] R.A. Fisher, H.G. Thomton and W.A. Mackenzie.
The Accuracy of the Plating Method of Estimating the Density of Bacterial Populations, with Particular Reference to the Use of Thomton's Agar Medium with Soil Samples.
Ann. Appl. Bot. 9:325-359, 1922.

[30] N.I. Fisher, T. Lewis and B.J.J. Embleton.
Statistical Analysis of Spherical Data.
Cambridge University Press, 1987.

[31] Glynn, E. F.
A Probabilistic Approach to the Stability of Rock Slopes,.
PhD thesis, M.I.T., 1979.

[32] N.H. Gray, et al.
Topological Properties of Random Crack Networks.
Math. Geol. 8:617-626, 1976.

[33] K.-H. Hanisch.
Reduction of n-th Moment Measures and the Special Case of the Third Moment Measure of Stationary and Isotropic Planar Point Processes.
Math. Oper. u. Statist 14:421-435, 1983.

[34] L. Heinrich.
On a Test of Randomness of Spatial Point Pattems.
Math. Oper. u. Statist. 15:413-420, 1984.

[35] L. Heinrich.
Asymptotic Normality of a Random Point Field Characteristics.
Statistics 17:453-460, 1986.

[36] W.G.S. Hines and R.J.O. Hines.
The Eberhardt Statistic and The Detection of Nonrandomness of Spatial Point Distribution.
Biometrika 66:73-79, 1979.

[37] Hoek, E. and J. W. Bray.
Rock Slope Engineering.
The Instn. Min. Metall., 3rd ed., 1981'.

[38] P. Holgate.
Some New Tests of Randomness.
J. Ecology 53:261-266, 1965.

[39] B. Hopkins.
A New Method for Determining the Type of Distribution of Plant Individuals.
Annals of Botany 18:213-227, 1954.

[40] P.H.S.W. Kulatilake.
State-of-the-art in Stochastic Joint Geometry Modeling.
Proc. 29th U.S. Symp. Rock Mech. :215-229, 1988.

[41] P.R. La Pointe.
A Method to Characterize Fracture Density and Connectivity Through Fractal Geometry.
Int. J. Rock Mech. Min. Sci. 25:421-429, 1988.

[42] P.A.W. Lewis and G.S. Shedler.
Simulation of Non-homogeneous Poisson Processes by Thinning.
Naval Res. Log. Quart. 26:403-413, 1979.

[43] J.C.S. Long, et al.
Porous Media Equivalents for Networks of Discontinuous Fractures.
Water Res. Res. 18:645-658, 1982.

[44] H.W. Lotwick and B.W. Silverman.
Methods for Analysing Spatial Processes of Several Types of Points.
J. R. Statist. Soc., B. 44:406-413, 1982.

[45] K.V. Mardia.
Statistics of Directional Data.
Academic Press, 1972.

[46] B. Matern.
Spatial Variation.
Springer-Verlag., 1986, 2nd Ed.

[47] R. Mead.
A test for Spatial Pattern at Several Scales using Data from a Grid of Contiguous Quadrats.
Biometrics 30:295-307, 1974.

[48] M.D. Mountford.
On E. C. Pielou's Index of Non-Randomness.
J. Ecology 49:271-275, 1961.

[49] J. Neyman and E.L. Scott.
Statistical Approach to Problems of Cosmology.
J. R. Statist. Soc. B 20:1-43, 1958.

[50] J. Ohser and D. Stoyan.
On the Second-order and Orientation Analysis of Planar Stationaly Point Processes.
Biom. J. 23:523-533, 1981.

[51] O'Reilly, K. J.
The Effect of Joint Plane Persistence on Slope Reliability.
Master's thesis, M.I.T., 1980.

[52] P. Parker and R. Cowan.
Some Properties of Line Segment Processes.
J. Appl. Prob. 13:96-107, 1976.

[53] L. Paterson.
Serrated Fracture Growth with Branching.
Proc. 29th U.S. Symp. Rock Mech. :351-358, 1988.

[54] E.C. Pielou.
The Use of Point-to-Plant Distances in the Study of the Pattern of Plant Populations.
J. Ecology 47:607-613, 1959.

[55] B.D. Ripley.
Modelling Spatial Patterns.
J. R. Statist. Soc. B. 39:172-212, 1977.

[56] B.D. Ripley.
Analyses of Nest Spacings.
In B.J.T. Morgan and P.M. North (editor), *Statistics in Ornithology*, pages 151-158. 1985.

[57] B.D. Ripley.
Point Processes for the Earth Sciences.
In C.F. Chung, et al. (editor), *Quantitative Analysis of Mineral and Energy Resources : NATO ASI*, pages 301-322. 1988.

[58] P.C. Robinson.
Connectivuty of Fracture Systems - A Percolation Theory Approach.
J. Phys. A 16:605-614, 1983.

[59] P. Segall and D.D. Pollard.
Joint Formation in Granitic Rock of the Sierra Nevada.
Geol. Soc. Am. Bull. 94:563-575, 1983.

[60] J. Serra.
The Boolean Model and Random Sets.
Com. Grap. & Image Proc. 12:99-126, 1980.

[61] Shair, A. K.
The Effect of Two Sets of Joints on Rock Slope Reliability.
Master's thesis, M.I.T., 1981.

[62] M. Shinozuka and C.-M. Jan.
Digital Simulation of Random Processes and Its Applications.
J. Sound & Vib. 25:111-128, 1972.

[63] B.W. Silverman.
Density Estimation for Statistics and Data Analysis.
Chapman & Hall, 1986.

[64] D. Stoyan.
Interrupted Point Processes.
Biom. J. 21:607-610, 1979.

[65] D. Stoyan.
Statistical Analysis of Spatial Point Processes : A Soft-core Model and Cross-corrrelations of Marks.
Biom. J. 29:971-980, 1987.

[66] D. Stoyan and J. Ohser.
Cross-correlation Measures of Weighted Random Measures and Their Estimation.
Theo. Prob. Appl. 29:345-355, 1985.

[67] D. Stoyan and H. Stoyan.
On one of Matern's Hard-Core Point Process Models.
Math. Nachr. 122:205-214, 1985.

[68] D. Stoyan, et al.
Stochastic Geometry and Its Application.
John Wiley & Sons, 1987.

[69] G.J.G. Upton & B. Fingleton.
Spatial Data Analysis by Example.
John Wiley & Sons, 1985 Vol. 1.

[70] D. Veneziano.
Probabilistic Model of Joints in Rock.
Internal Report, M.I.T., 1979.

[71] K. Watanabe.
Stochastic Evaluation of the Two Dimensional Continuity of Fractures in a Rock Mass.
Int. J. Rock Mech. Min. Sci. 23:431-437, 1986.

[72] P.A. Witherspoon and J.C.S. Long.
Some Recent Developments in Understanding the Hydrology of Fractured Rocks.
Proc. 28th U.S. Rock Mech. :421-432, 1987.